ADVANCED INTELLIGENT SYSTEMS
Open Access
www.advintellsyst.com

Artificial Intelligence in Process Engineering

Christoph Thon, Benedikt Finke, Arno Kwade, and Carsten Schilde**

In recent years, the field of Artificial Intelligence (AI) is experiencing a boom, caused by recent breakthroughs in computing power, AI techniques, and software architectures. Among the many fields being impacted by this paradigm shift, process engineering has experienced the benefits caused by AI. However, the published methods and applications in process engineering are diverse, and there is still much unexploited potential. Herein, the goal of providing a systematic overview of the current state of AI and its applications in process engineering is discussed. Current applications are described and classified according to a broader systematic. Current techniques, types of AI as well as pre- and postprocessing will be examined similarly and assigned to the previously discussed applications. Given the importance of mechanistic models in process engineering as opposed to the pure black box nature of most of AI, reverse engineering strategies as well as hybrid modeling will be highlighted. Furthermore, a holistic strategy will be formulated for the application of the current state of AI in process engineering.

1. Introduction

The field of process engineering deals with a broad spectrum of subordinated fields like mechanical-, thermal-, bio-, electrochemical-, chemical-, and systems-process engineering as well as nanotechnology. Moreover, there exist many interdisciplinary overlaps to additional fields like mechanical engineering and material science. As a consequence, the challenge arises to model the underlying process principles in a parameter-space of often very high dimensionality with highly complex dependencies between different parameters, whose modeling is necessary for the subsequent process design, optimization, control, and fault diagnosis. Usually this challenge is addressed by experimentation and simulations involving a high degree of expensive manual intervention in modeling. Therefore, the objective arises to find and integrate methods which allow the reduction of experiments and simulations applied for the modeling to a necessary minimum. In addition, due to the exponentially increasing availability of sensors and networks, in many fields large amounts of data are produced, which are too extensive and too complex for manual modeling. Increasingly, the need arises for an automated way to model complex systems.

A promising solution for this problem is the incorporation of Artificial Intelligence (AI) and more specifically Machine Learning (ML) techniques in the modeling afford. The field of AI has a long history comprising different stages of varying enthusiasm followed by disappointment. However, the emergence of the subordinate field of a ML and especially the advent of Deep Learning (DL) allowed the field to finally live up to the earlier expectations and to promote progress in all kind of fields, resulting in an increasing number of practical applications and increasing usability.[1] In the field of process engineering and to a broader extend also in mechanical engineering various academic and industrial implementations of AI and ML techniques have been demonstrated with the goal of facilitating and simplifying progress in the field. The objective of this Review consists in providing an overview of already existing use cases, in giving an overview of the applied techniques, methodologies, and strategies as well as the goals with respect to process engineering. In addition, acompanying means are addressed as well as the overall procedure for projects, which combine process engineering tasks with the application of AI in regard to predictive modeling as well as to deriving mechanistic models. The underlying fields are discussed, and practical applications are presented, to structure the goals as well as the used methods of AI. Relevant preprocessing steps are discussed, as well as the application of hybrid models to facilitate the AI progress and subsequent postprocessing steps. Finally, a holistic framework for the application of AI and ML techniques for process engineering projects is proposed, addressing the goal of building predictive models as well as the derivation of mechanist models.

2. Process Engineering

The field of process engineering in whose context the application of AI is discussed in this Review deals with the transformation of raw materials into commercial products, to be further used in related industries like the manufacturing industry where these intermediary products and further products are finished to discrete products.[2] Directly linked subsequent fields of activity in

C. Thon, B. Finke, Prof. A. Kwade, Prof. C. Schilde
Institute for Particle Technology (iPAT)
Technische Universität Braunschweig
Volkmaroder Str. 5, Braunschweig D-38104, Germany
E-mail: c.thon@tu-bs.de; b.finke@tu-bs.de; a.kwade@tu-bs.de; c.schilde@tu-bs.de

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/aisy.202000261.

DOI: 10.1002/aisy.202000261

Table 1. Fields of activity in process engineering with important respective tasks and achievement; reproduced and modified from Grossmann et al.[3]

Process Design	Process Operations
Design of energy and energy recovery networks	Scheduling of process networks
Design of aggregates	Multiperiod planning and optimization
Design of aggregate networks	Data reconciliation
Hierarchical decomposition flow sheets Superstructure optimization	Real-time optimization
Design multiproduct batch plants	Flexibility measures
Process Control	**Supporting tools**
Model predictive control	Sequential modular simulation
Controllability measures	Equation based process simulation
Robust control	AI/Expert systems
Nonlinear Control	Large-scale nonlinear programming (NLP) Optimization of differential algebraic equations
Statistical Process Control	Mixed-integer nonlinear programming (MINLP) Global optimization
Process Monitoring	Sequential modular simulation
Thermodynamics-based control	Equation based process simulation

process engineering are "process design," "process control," "process operations," and "supporting tools," as shown in **Table 1**.

In addition, process economics may be added to this list, overlapping with the engineering tasks. For each of these fields a multitude of subsequent subfields and derived tasks exist.[3] For instance, derived tasks encompass life cycle management, process assessment and risk management. The basic concept of process engineering resembles the concept of chemical engineering, established in 19th century, but applied to all kinds of industrial processes that transform matter and energy like pharmaceuticals, biotechnological substances, paper, cement, metal, cosmetics, food, chemicals, and others.[2] Also included is Process Systems Engineering (PSE). Table 1 shows the main fields of activity in process engineering, with some examples of subtasks and recent achievements in PSE.[3] Process engineering operates on various length and timescales, showcased by the chemical supply chain in **Figure 1**. All of the described aspects within the supply chain are linked with specific models, representations, experiments, simulations, unit operations, process parameters, and conditions.

Similarly, spatial and temporal spectra exist for the other subfields of process engineering. The mayor challenges of process engineering are the development of new products, unit operations, predictive capabilities, multifunction integration, means for cost- and complexity reduction, closer integration of the mentioned fields of activity, the development of plant wide control- and predictive control systems, the integration of business planning, supply chain management, logistics, innovative plant design, modeling environments of increased flexibility, multiscale modeling, scheduling, the increasing amounts of sensors for measurement, life-cycle modeling, etc.[3] In all of these levels new challenges arise, paired with large and increasing amounts of data, which demonstrates the high potential for the application of self-organizing, data-driven ML-tools in process control, optimization, fault detection as well as for modeling.

3. Artificial Intelligence

To clarify the terms used in this Review, it is important to define the terminologies in the field of AI, to discriminate the most important concepts and to give some overall overview of the fields' background. The field of AI established itself in the 1950s and 1960s, with AI generally applying as an umbrella term to all machines displaying "cognitive functions," usually associated with human mind and behavior such as learning and

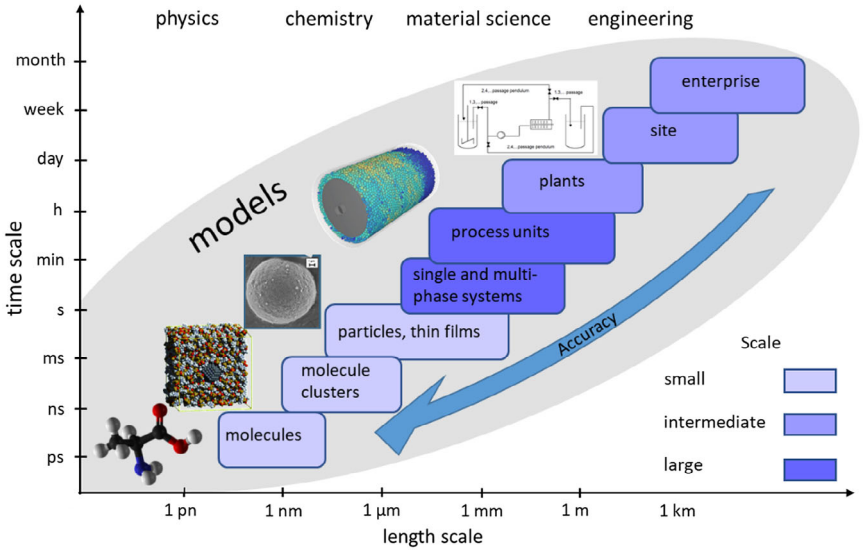


Figure 1. Chemical supply chain at the example of particle technology demonstrating the broad spectrum of relevant length and timescales. Second image from below: Reproduced with permission.[101] Copyright 2011, Wiley.

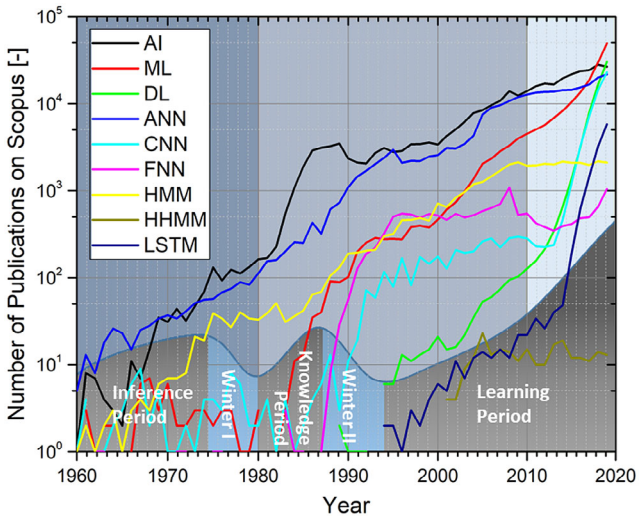


Figure 2. Quantitative number of Scopus citations of different ML methods over time in conjunction with a qualitative temporally corresponding representation of respective AI epochs at the lower part, for full terms see the abbreviation index, Supporting Information.

problem solving. As then different subsections evolved, with the most recent and prominent subfields being ML and its respective subfield DL. ML summarizes all techniques in which computers are capable of learning without being explicitly programmed by being able to learn from and make predictions on data.[4] Furthermore, "DL is the study of Artificial Neural Networks and related ML algorithms that contain more than one hidden layer."[5] Here, hidden layers describe layers of neurons between input and output neurons, with their number determining the complexity of the trained model. For further details, see Section 5. DL encompasses ML algorithms (mostly but not exclusively Artificial Neural Nets [ANNs], see Section 5) which consist of a cascade of layers, to recognize and extract higher-level features on the basis of lower-level ones.[4] In **Figure 2**, the number of Scopus publications regarding the three categories as well as of some prominent AI techniques (AI, ML, DL) are quantitatively plotted, showcasing especially the relatively recent rise of terms associated with ML and DL. The respective AI techniques are later explained in chapter 3.2.2. Also included as a color-coded graph in the lower part of Figure 2 is a qualitative representation indicating the level of prominence and enthusiasms over time in the field, also including the relevant historic periods printed in white. Especially noteworthy is the steep ascent of mentioning of ML, DL, and ANNs in the time since 2010, falling into and giving rise to the "Learning epoch."

In **Figure 3**, the concepts of AI, ML, and DL as well as their dependencies are shown. Methods dedicated to the respective paradigms are also listed together with application examples.

In Figure 2, the "inference period" marks the beginning of the field, starting with the famous 1956 Dartmouth conference, where the concept of AI was first proposed.[1] The first neural nets, like the perceptron, were limited in their usefulness due to their one-layer structure which caused it to fail at solving even rather simple linear inseparability problems.[6] Due to this and overoptimistic estimations of the speed of initial progress, AI experienced a first AI-winter. Subsequently, it experienced a second boom phase, the "knowledge epoch," in the 1980s with the advent of multilayer neural nets and back propagation (BP). Remaining mathematical problems restricting the number of layers in the nets in conjunction with problems like poor generalization and difficulties in the interaction of AI with the environment lead to the second AI winter around 1995. After a relatively short second winter, the third epoch, the "learning epoch" started caused by the solution of the mathematical problem by a team around Hinton[7–9] and supported by the rise of Big Data[1] Consequently, Deep Neural Nets (DNNs) with vastly more hidden layers could be trained without falling into local minima, gave rise to AI models of increasing complexity covering patterns of increasingly higher order like faces building on lower-level patterns like simpler geometrical patterns. The training and inferencing of increasingly complex neural networks was additionally enabled by the exponential growth of data and computing power. This exponential growth, which was first discovered by Gordon Moore, became known as Moore's law. Moore realized that the number of transistors fitting onto an integrated circuit were

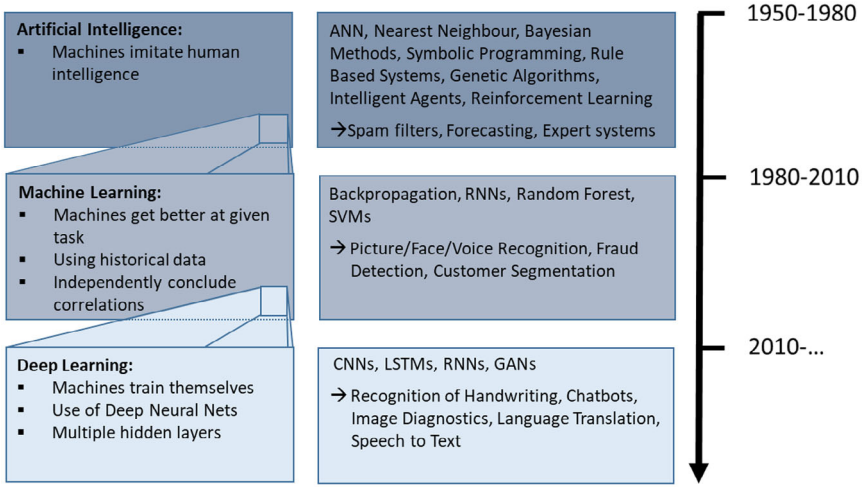


Figure 3. Paradigms of Artificial Intelligence and related usecases.

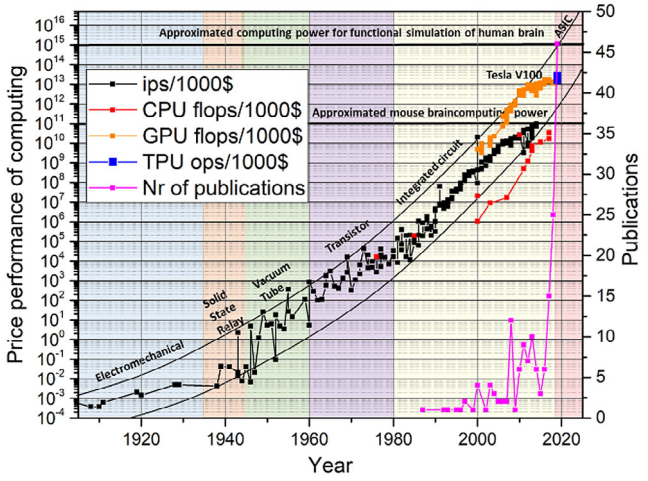


Figure 4. Exponential increase in price performance of computing over 120 years and increase in publications of *ML* in the context of mechanical engineering. Based on Kurzweil et al.[11], modified and extended with independent data evaluation with data from ref. [11–14].

doubling every year and predicted this trend to continue, later he adjusted the doubling time to 18 months.[10]

On a longer time frame (see **Figure 4**), as analyzed by Kurzweil,[11] the overall increase in computational price performance is even following an exponential trend on two levels with a growing exponent and a therefore decreasing doubling time, with a smooth development independent of the current type of underlying computer hardware like vacuum tubes or transistors.[11] The graph was updated and modified by the authors of this Review, the number of iterations per second is shown in black dots, for more recent hardware in flops, red for CPUs, orange for GPUs, and blue for TPUs (application specific integrated circuits [AI ASICS]).

The last points in blue are marking the Tesla V100 and the Google Edge TPU (4 Tops DL-Inference for around 100$), both first-generation AI-ASICs. As integrated circuits have not been the first but the fifth major computer paradigm, it is however likely that another sixth paradigm will continue the exponential increase in price performance and capacity of computing, with Intel already having announced its Foveros 3D-circuit architecture,[12] Google demonstrating a functional 53 qubit quantum computer[13] as well as the ascent of cloud computing and further potential successor technologies demonstrating their capabilities. In case of AI-ASIC's, their appearance is a very recent development. For the given two examples in Figure 4, the price performance is 16 Tops/1000$ for the Tesla V100 and 27 Tops/1000$ for the Google Edge TPU.[14] Other even more powerful ASICs like the Google TPU 3.0 with 420 TeraFLOPS[15] can be accessed in the cloud. In 2020, the company Cerebras announced its Cerebras ASIC with 57 times the transistor count and surface area[16] as a Tesla V100 with 130 TeraFlops, the exact computing power is not yet known. Both the Cerebras ASIC and the Google TPU v3 cannot be priced, wherefore they are not included in Figure 4. However, the recent start and quickly accelerating speed in ASIC-development show the near term potential for further gains in price performance.

The significant increase in computing power demand and availability was investigated by Amodei et al., who analyzed the total amount of computing, in petaflops-days dedicated to specific popular examples of recent AI milestones over the years. This analysis revealed a doubling time of spent petaflop/s-days every 3.4 months, resulting in a total increase in a factor of 300 000 between the AlexNet in 2012, the initial kick off of the DL revolution and 2018.[17] Furthermore, a study conducted by Hermandez et al. investigated the progress in the actual software efficiency of ML algorithms, independent of hardware capability and analyzed the applied computing power required to achieve a set performance with the respective latest algorithms of different years. Observing various algorithms of recent years, they measured doubling times in terms of efficiency for image recognition on the basis of ImageNet between 16 and 17 months, between 4 and 6 months for translation tasks, and even shorter doubling times in game playing with 4 months for the game of Go and just 25 days for the complex PC-strategy game Dota.[18] In addition, Hermandez et al. found the overall progress resulting from these gains to massively increase the effective available training compute, in other words, the gains in overall efficiency as a consequence of the overall progress in Moores law, software efficiency, and in spending and parallelization to train current AI problems of interest in comparison with a simple scale up with the means of 2012. The approximated the increase to be by a factor of 7.5 million in the remarkably short timescale between 2012 and 2018.[18]

They took this to be the result of increases in software efficiency and the previously discussed hardware capabilities analyzed by Amodei,[17] caused by the gains in overall efficiency (Moores law), software efficiency, and in spending and parallelization (**Figure 5**).

It was not addressed to what degree these trends could be extrapolated, however it was noted, that similar trends of long-term exponential progress in the efficiency of software independent of hardware gains exist in other domains, like a continuous doubling in the efficiency of mixed integer programming over

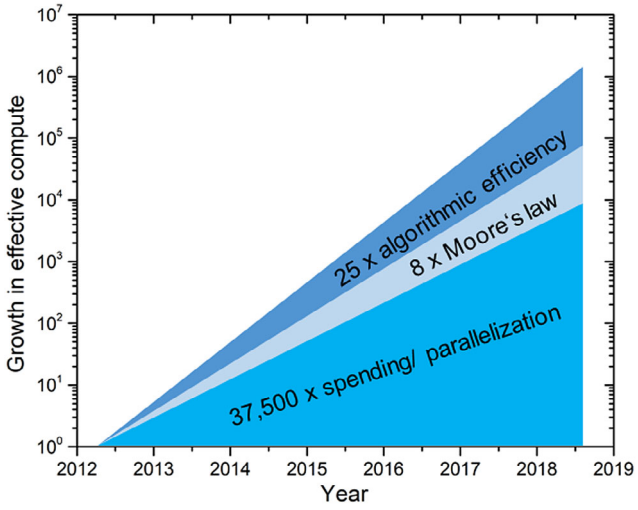


Figure 5. Approximated growth in effective compute between 2012 and 2018 with respective factors.

21 years with a doubling time every 13 months.[19] The same holds for other examples from game programs, mixed integer programs, physics simulations, and SAT solvers, as shown by Grace.[20]

Another important factor for AI is the advances in AI-platforms. While the first neural nets were implemented manually on the basis of usually self-written codes, more recently, a multitude of ML frameworks and libraries emerged, with the most relevant being Tensorflow, Keras, Caffe, Caffe2, Theano, Lasagne and Blocks, MXNet, CNTK, Torch, PyTorch, Pylearn2, Scikit-learn, Matlab including MatconvNet Matlab DL, DL tool box, Chainer, and Deeplearning4j.[21] A detailed description of these platforms together with their advantages and disadvantages and respective benchmark tests for comparison was presented by Wang et al.[21] The frameworks are set up in a way to enable the application of advanced ML without the need for profound knowledge regarding the implementation of ML algorithms or software engineering. Within these ecosystems additional tools are implemented, like tools for visualization, data validation, preprocessing, the analysis of build models as well as serving. In case of TensorFlow for instance with TensorFlow Hub a tool exists for the reuse of pretrained neural nets.[22] Higher-level API's, building on top of lower level ones facilitate the implementation and application of ML even further. Keras for instance focuses on DL, being therefore less flexible but easier and faster to implement. It allows to access the full potential and range of possibilities of the powerful underlying libraries while providing a high-level API. Taking Keras as an example, with its guiding principles of construction of modularity, minimalism, extensibility, and the use of Python as easy to use programming language, the increasing accessibility of AI frameworks, platforms, and libraries is shown.[23] In addition to the already heavily facilitated frameworks, libraries, and APIs, since 2016, the concept of Automated ML (AutoML) further simplifies the application of ML by automating the optimization effort in finding the best ML-architecture and hyperparameters. Hyperparameters refer to user-defined parameters, which govern the model behavior. Their value is not adjusted by training of the AI. Here, the design of the network is conducted iteratively using methods like Deep Reinforcement Learning, genetic algorithms (GAs), or gradient-based methods (see Section 5) and network morphism. Often AutoML algorithms are cloud-based and open-source variants like AUTO-KERAS are also available.[24]

4. AI Applications in Process Engineering

As previously shown in Figure 4, the amount of publications regarding the application of AI algorithms in mechanical and process engineering has increased exponentially in recent years. Based on a literature survey of these publications, the described applications in process engineering can be classified to the following main objectives: data rectification, predictive process modeling, process optimization, fault/anomaly detection, process control and the derivation of mechanistic models, as schematically shown in **Figure 6**.

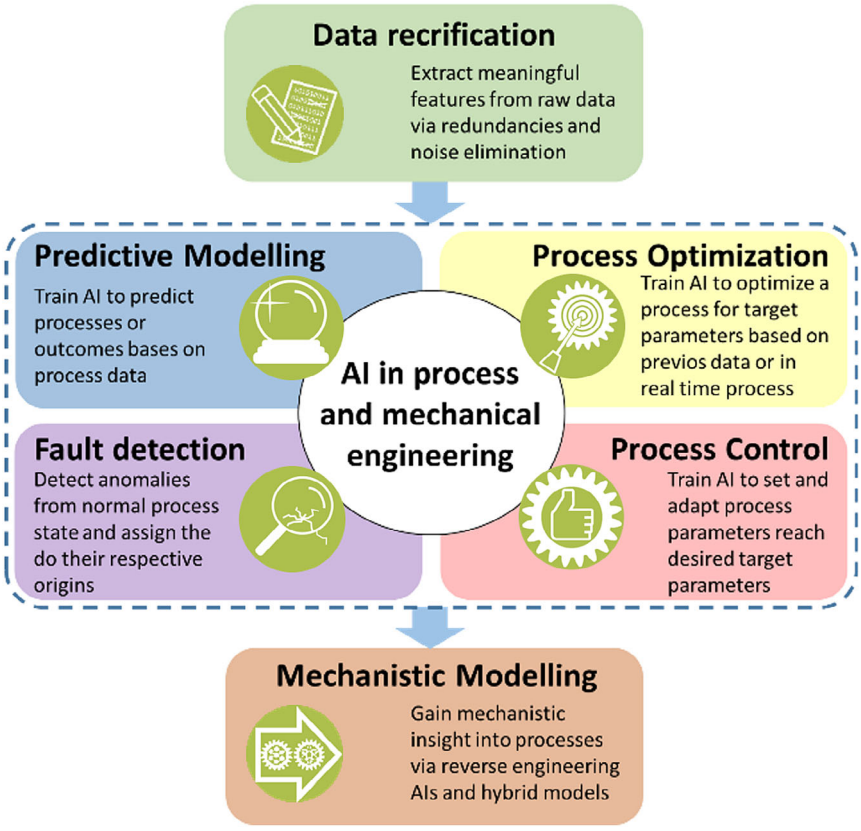


Figure 6. Overview of the main applications of AI in process engineering and related fields.

Of these 6 objectives *predictive process modeling, process optimization,* and *control* and *fault diagnosis* are directly located in the overlapping realm of process engineering and AI. *Data rectification* is a preceding process with the goal of data preprocessing, which is often used before the actual AI training. Predictive modeling is the most prominent objective which is also often involved in the other applications, mostly as an intermediary or preliminary step in the overall procedure (e.g., as a basis for subsequent process optimization, process control, fault diagnosis, or mechanistic modeling). *Mechanistic modeling* occupies a special position within the framework of objectives, as its main target lies in breaking down the trained empirical data-driven models toward transparent mechanistic or physical white box models and insights. This encompasses the use of reverse engineering strategies of the preliminary trained black box models. To facilitate mechanistic modeling, hybrid models or gray box models are also applied. In Section 6, data rectification in the larger context of preprocessing, the application of hybrid models as well as mechanistic modeling are described in more detail.

In **Table 2**, examples for the use of ML for the respective process engineering tasks mentioned in Figure 6 are shown together with the specific goals, the input and output data used for training the network, as well as the source of data or the coupled experimental set-up in cases in which the AI was trained and used in a closed loop. In some cases, multiple AI methods were used in direct succession with different purposes, e.g., a feature extraction algorithm, which identifies characteristic patterns in large amounts of data (clustering, see Section 6.1). By applying multiple AI methods, the carved out (in technical terms: compressed) features could act as the input parameter for the training of a predictor neural net. For the purpose of clearer referencing of the respective subtasks in subsequent chapters of this publication, the literature sources are assigned with an additional ID for internal assignment. Subsequent subtasks in one of the referenced papers, which build upon each other chronologically (meaning the successive application in a specific order is of significance), are hereby reference by x.1, x.2, etc., whereas the display of multiple independent AI applications in one literature source which do not build upon each other are listed by x.*a*, x.*b*, etc.

In the following, the most significant examples of the aforementioned case studies are described in more detail with the focus on selecting profound features and methodological strategies to give readers a roadmap to adapt and transfer these to other usecases within the realm of process- and mechanical engineering as well as in other engineering domains.

Predictive Modeling: An example for the use of ML in process engineering as a predictor model was shown by Geng et al.[25] They used a two-step approach to train an ANN to predict a chemical reaction process in a complex setup encompassing a column, a reboiler, and a reflux tank, as shown in **Figure 7**.

In the investigated process pure terephthalic acid is produced in a plant consisting out of a solvent dehydration column, a reflux tank and a reboiler. A total of 17 input parameters like water return flow, feed flow and composition, further flow rates, temperatures in different plant positions and the reflux tank level was used with the acid content leaving the plant as output parameter. Before the actual training of the ANN as a predictor, a preliminary Auto Encoder (AE) (see Section 6) was integrated to extract the main features (also called dimensionality reduction) out of the training data. Through the reduction of redundancies and noise, the efficiency and stability of the subsequent AI-modeling was increased. Based on the detected features and the acid content data the ANN, an Extreme Learning Machine (ELM), was trained to accurately predict the outcome of the process. The ELM is characterized by the random setting of the parameters of the hidden layers in the interest of avoiding local minima in the gradient decent process. In total 259 training samples were used, 172 for training and 87 to test and evaluate the accuracy of the trained predictor. The trained AI model showed good generalizability with the combination of an AE with an ELM. It demonstrated a higher level of generalizability than the sole use of the ELM. The resulting mean relative error for the combined AE-ELM was between 0.3% and 0.6% for different numbers of hidden layers (hidden layers see Section 5), and around 3% for the sole ELM.[25] This example proves that the behavior of a given physical process can be modeled predictively by training an AI system like a neural net with historical plant data. With pre-existing knowledge of the inherent causalities, the modeling can be facilitated by the intelligent allocation of parameters as input and output data, as will be later described in Section 6 regarding supervised learning. That trained models of good generalizability allow the prediction of process behavior even for previously not investigated parameter settings to cut costs of extensive systematic experimental investigations. Furthermore, it can be used as the basis for subsequent process optimization, control, anomaly detection and also to derive mechanistic insights.

Process Optimization: Zhang et al.[26] demonstrated process optimization tasks using ML capabilities by two examples. In the first case, a hybrid model (see hybrid models in Section 6) was trained for a continuous stirred tank reactor to represent a reversible exothermic chemical reaction process. Here, the ANN was trained to replace the first-principle reaction rate equations within the mass and energy equations, which would otherwise be difficult to obtain without proper knowledge of the underlying reaction mechanisms. The input parameters influencing the reaction rates were the concentrations of the substances A and B and the temperature. The prediction of the reaction rates was trained on the basis of 8 million data points. The trained networks achieved a mean square error of less than 10^{-7}.

As a second example, an ANN was trained to replace the phase equilibrium functions in a distillation column, as shown in **Figure 8**.

Here, in a first-principles approach, the mole function was calculated and the ANN was trained with the vapor-phase mole fraction as input and equilibrium temperature T and liquid-phase mole fraction y as the two output parameters. About 1500 training datasets of T, x, and y were generated with the Aspen property library. For the ML, two layer Feedforward Neural Nets (FNNs) (see Section 5) were used, the mean squared error was around 10^{-07}. On the basis of these two hybrid models Real-Time Optimization (RTO) and Model Predictive Control (MPC) cost functions were established to maximize economic productivity with the trained neural nets replacing the respective nonlinear first-principle functions in the RTO and to set the optimal set points for the controllers. In the RTO of the stirred tank

Table 2. Listing of ML example within the context of process engineering and mechanical engineering in dependence of the respective task within the process engineering field extended by reasonable examples from mechanical engineering (printed in italics) with additional information regarding input and output data and the source of training data.

		ID	Goal	Input	Output	Source/Counterpart
Rectification 	[25]	*1.1*	*Compress sensor data to meaningful features (Step 1)*	*17 Process parameters (water return, feed flow, composition, ...)*	*Features*	*Source: Production samples*
	[27]	*2.1*	*Compress laser-welding sensor data to features (step 2)*	*Sensor data (Laser Welding pictures)*	*Features*	*Source: Pictures of laser welding*
Prediction Modeling of process (outcome)	[36]	*3.a*	*Predict robot behavior*	*Angle positions/velocities of three links, torque of 2 joints*	*Angular velocity of robot arm*	*Source: Pumadyn family of data sets (8 and 32 nm)*
	[36]	*3.b*	*Predict motor temperature*	*Temperatures, motor behavior, current, speed and torque*	*Motor Temperature*	*Source: Electric motor temperature data set (Simulation)*
	[25]	*1.2*	*Predict plant production*	*Plant features (derived in 1.1)*	*Acetic acid content*	*Source: Production samples*
	[90]	4	Identify textures of materials from shaping, milling, sand blasting, EDM			
	[91]	5	Predict Aspect ratio caused by hydraulic fracturing			
	[92]	*6*	*Prediction of tensile strength based on welding process parameters*	*Rotational speed and travel speed*	*Tensile strength*	
	[27]	*2.2*	*Make predictions of current performance of laser welding*	*Features (derived in 2.1)*	*Labels from experts)*	*Source: Sensor data from laser welding*
Process Optimization 	[26]	7.a	Prediction of reaction rate equations in tank reactor – optimize operation profit (RTO) and set optimal controller set points (MPL)	Kinematic factors (CA, CB,T)	Reaction rate *r*	Validation: Profits/costs-comparison of simulations and first-principle approach
	[26]	7.b	Predict vapor-phase equilibrium l iquid phase mole function in distillation column and optimize operation profit (RTO) and set optimal controller set points (MPL)	Vapor-phase mole fraction (x)	T and liquid-phase mole fraction (y)	Source: Aspen property library data sets Validation: Profits/costs-comparison of simulations and first-principle approach
Proc. Control 	[27]	*2.3*	*Optimization of laser welding process*	*Features from simulation (derived from simulation with trained system from 2.1)*	*Labels (derived from simulation from 2.2)*	*Setup with feedback (Simulation)*
Fault/anomaly detection 	[29]	8.a	Classification of plant anomalies caused by cyberattacks in chemical reaction patterns			Source; (Additional Tennessee Eastman Process simulation—Simulation)
	[29]	8.b	Classification of pipeline anomalies caused by cyberattack			Source; (Gas pipeline dataset—Simulation)
	[30]	9	Classification of faults in a grinding process	30 Input features (24 with dimensionality reduction)	8 Labels (7× faults, 1× normal)	Experiments

reactor cost functions for reactant conversion, and heat cost were established and optimized regarding the total cost, regarding the trained AI models for the reaction rates. Similarly, for the distillation column, an objective function for the profit was developed, depending on process for products, feed, and energy. In both cases, hybrid models were established based on established

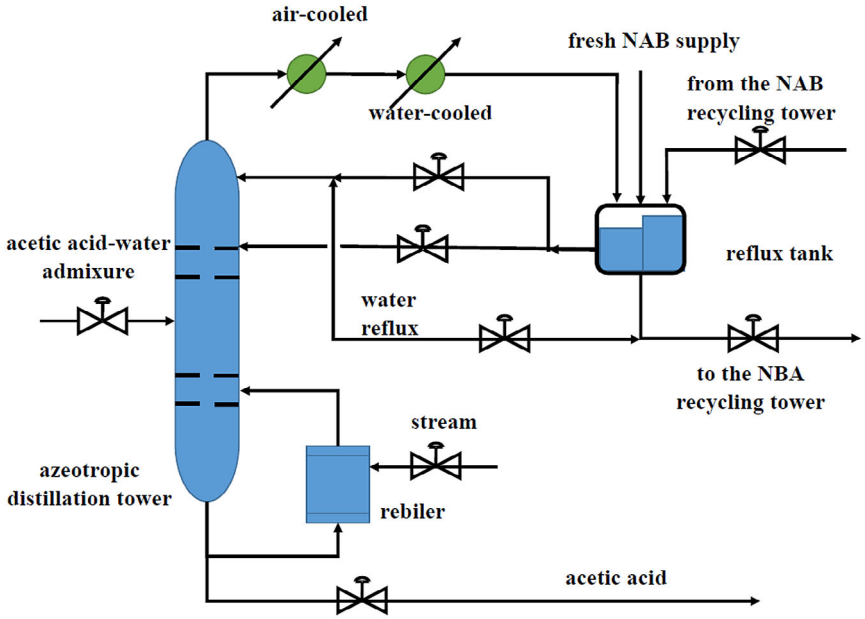


Figure 7. Dehydration column scheme. Reproduced with permission.[25] Copyright 2019, IEEE.

models and formulas to simulate the respective chemical processes. Neural nets were applied where parameters were difficult to model. These predictive systems were then embedded in a superordinate system of cost function to be optimized. On the basis of the respective RTOs, MPC strategies for process control were established, which will be discussed in the following subchapter on process control.[26] Process optimization can be accomplished with hybrid models or the use of ML models without a first-principle combination. In addition to the predictive model, a target value is needed, embedded in a cost function or a system of cost function. The optimization can be applied on the basis of a fixed data set or dynamically by generating new data in the process, in the latter case, usually also involving process control, described in the respective subchapter. This dynamic approach involves the ML school of reinforcement learning, see Section 6.

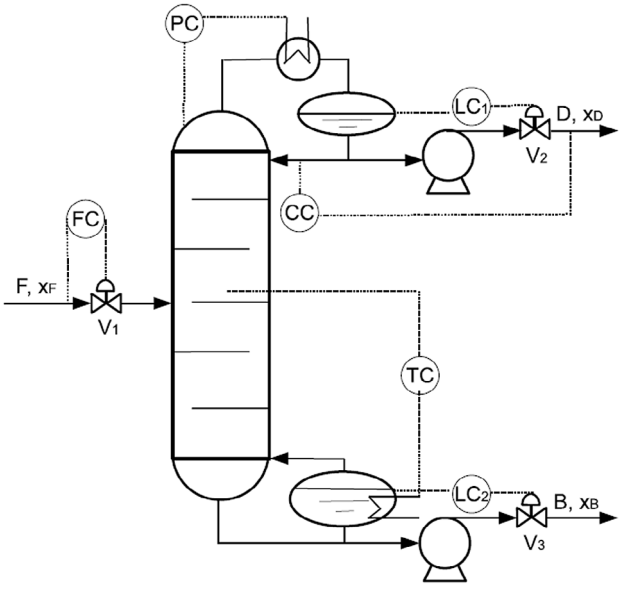


Figure 8. Distillation column schematic with controls. Reproduced under the terms of the CC BY 4.0 license.[26] Copyright 2019, MDPI.

Process control: As described in the previous subchapter on process optimization, Zhang et al.[26] established predictive hybrid models for a stirred tank reactor and a distillation column with subsequent process optimization. On the basis of the RTOs containing hybrid-AI models, individual MPC strategies were derived. In this case, the process control was set up in conventional ways, with a Lyapunov-based model predictive controller, which describes a model predictive controller to adjust the optimal input trajectories to set the process parameters to the steady-state predicted by the RTO. For the dehydration column, the optimal parameters from the RTO are passed to six controllers (with fixed values for flow rate, pressure and two-level controllers and the RTO-based values for the temperature and the concentration controller) to set the right process conditions.[26] In these two cases, the AI was used for predictive hybrid modeling of the underlying physical process, with subsequent optimization and control strategies using conventional approaches instead of AI. Alternatively, the entire process control can be implemented via means of AI. For instance, Günther et al.[27] demonstrated the process automation using ML in a multistep approach consisting of preliminary feature extraction, predictive learning, and derived control. All three steps were carried out using DL, however, the exact methodology differed depending on the respective task. The investigated case was a laser welding process. In the first stage, an AE (see Section 6) was trained with 16 000 laser welding images to train the system to extract and compress the main features into an abstracted more generalized transformation-invariant representation of lower dimensionality.

The pictures were divided to 4×4000 pictures for fourfold cross validation. A subsampling of regions in the pictures to 32×32 was carried out to normalize the input data. For 16-dimensional output features, the trained AEs reached a mean reconstruction error of 16.6%. In the next step, a predictor was trained to evaluate the performance of laser welding. Here, an approach called "nexting," an ML technique to make short-term predictions about the immediate future of a process, was implemented to estimate the outcome based on real-time signals. Input parameters were the extracted signals from the AE from step one. With the 16-dimensional feature space from the AE plus the photodiode data, a 19-dimensional vector space resulted for the input data. The features were labeled by human experts to classify the quality of the process, with the labels (1–4) being the output parameters (Supervised Learning, see Section 6). The predictor was evaluated with new data, not used in training to prove the system's ability for generalization. After training, the distance error for the distinction between good and bad classes could be reduced to a value of 0.28. For the final step of process control, a continuous action Actor-Critique-Reinforcement-Learning (ACRL) technique was used (see Section 6) with the actor learning to generate appropriate actions and the critic learning to evaluate the performance of the actor. The actual process control took place in a closed loop with a laser welding simulation, with the laser width as the control parameter of the simulated laser welding system. The laser width was then processed with the DNN from step 2 to derive an evaluation of the current performance. The output action was the applied laser power. The evaluation of the trained neural net took place by evaluating, whether the desired welding depth was reached. The performance increased with the number of learning epochs. In total, 30 epochs (runs in the simulator) were carried out, achieving a negligible absolute welding depth error.[27] As mentioned, tasks regarding process engineering can be approached individually but also serially as closely integrated systems addressing more than one of the mentioned core tasks. Shastri et al.[28] demonstrated a process control strategy on the basis of preceding fault detection at the usecase of a three-tank system interconnected by two pipes and two pumps, with the goal of controlling water levels.[28] Training and testing data for normal and faulty conditions were generated within a simulation. In the simulation, two PI controllers acted as input and random noise was added. Resulting pump flow rates for given simulation inputs were chosen as indicator for process conditions, labeling the input data (1 for a completely open "leak" and 0 for a completely clogged valve). About 105 data points were generated, 19 for normal operation, the rest for five different types of fault. On this basis, a C4.5 classifier was used. This refers to an ML technique for the formation of a decision tree (DT) based in an iterative approach on the basis of a historical set of training data. The trained tree was evaluated by testing with an independent testing data set, not used for training, with a 92% correct classification rate. The trained decision was then translated into a rule set, effectively extracting knowledge from the historical data. In comparison with neural nets, this approach has the advantage of being more transparent (for model transparency see Section 6). The rule set was implemented into an expert system shell with if-else rules, with outputs being for instance the call for a human operator.[28] Similarly, such a rule set could directly be translated into control parameters for the plant to directly add control measures in case of systems failure.

The aforementioned examples show, that process control can be implemented by the single use of AI or in combination with conventional approaches. As shown in Sections 5 and 6, there are many variations of AI and approaches to use it. Dependent on whether the AI can be trained during operation or on the basis of historical data as well as its amount and labeling, the training can be operated supervised or unsupervised, or in form of reinforcement learning. A pretraining (see Section 6) in a safe environment like a simulation can precede the actual training in a physical environment. Combinations with existing control strategies can also be implemented and facilitated in the form of hybrid modeling (see Section 6). It is therefore generally recommendable to clearly identify the desired goals and to analyze the accessibility to the process parameters and to data. Furthermore, it is advisable to search for applications similar to the respective usecase.

Fault/anomaly detection: Sokolov et al.[29] investigated the use of ML to detect process anomalies of sensor readings and controller settings as an indicator for targeted cyberattacks in industrial plants. One example featured a chemical reaction plant consisting of a reactor, a condenser, a vapor–liquid separator, a compressor, and a stripper using the "Tennessee Eastman process simulation data for anomaly detection evaluation." Dataset with eight chemical components (A–H) were involved, along with the target components being G and H, shown in **Figure 9**.[29]

About 41 variables (like temperatures and pressures) were defined to describe the process at any given time. Twelve additional variables to control the process were included as well. The investigation used pregenerated data. Fifty-three features and 20 classes belonging to certain types of anomalies or to normal operation procedures were included. In total, a sample size of 5 250 000 records was used and divided into 70% for training and 30% for testing. Different methods (see Section 6) of ML were used, including Logistic Regression, Lasso, SVM, DT, Adaptive Boosting, Gradient Boosting, Random Forest, and a fully connected Deep Neural Network of which the latter method achieved the best result for detecting and classifying attacks. It

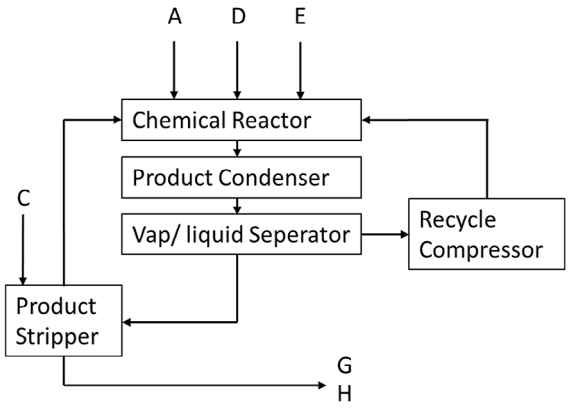


Figure 9. Tennessee Eastman process simulation. Reproduced under the terms of the CC BY 4.0 licence.[29] Copyright 2019, Faculty of Mechanical Engineering, Belgrade University.

achieved an accuracy of 0.82, followed by Random Forest (0.67), SVM (0.57), Gradient Boosting, Logistic Regression and Lasso (all 0.46), Adaptive Boosting (0.34) and DT (0.23). To further increase the accuracy and robustness the authors suggested to break down the process into subprocesses to decrease the complexity of the problem for the ML system, an approach closely related to hybrid models as described in Section 6.[29] As an additional example Xing-yu et al.[30] trained a fault recognition network for a grinding system with a 30-dimensional input feature data set consisting out of 1000 sets of feature vectors (864 of fault free and 135 sets of faulty data) while using and comparing a standard Back propagation network (BPN), a combined AE-Softmax network and an RNN-LSTM network (see Section 6), with the later one showing the most accurate performance. Thirty-dimensional data for the input parameters were used and compressed (see Section 6) to 24-dimensional data, the normal state together with 8 types of failure were chosen as output parameters. Accuracies of 93.3% for real faults and of 94.83% for nonfaults could be reached, the traditional BP network reached 56.7% for fault and 69.9% for free-fault predictions and the AE-Softmax network 82.2% for fault and 90.3% for free-fault predictions.[30] As will be discussed in Section 5, there are multiple categories of the application of AI-like regression, classification, clustering, and dimensionality reduction. On the field of fault detection, it is also used for the classification and clustering of faults.[30] The actual implementation depends on the existence of labeled or unlabeled data. Furthermore, it could be trained and executed on the basis of fixed data sets or in a dynamic process environment or a simulation, in the form of supervised, unsupervised, semisupervised, or reinforcement learning. Related to a specific usecase, the exact type of fault detection depends on the purpose, the available date, the way of integrating the AI into the process, etc. Fault detection in the broadest sense requires process data containing patterns of faults, and usually additional normal process data. The presence of labeled data facilitates the training.

5. AI Systematics and Techniques

As a consequence of the long history of AI and the increasing interest and practical real-world applicability of ML and DL, a broad spectrum of AI techniques has evolved. In this chapter, a general classification of AI is provided, the general functionality of Neural Nets is explained, and the most important variants of AI-algorithms are briefly presented with their respective advantages and disadvantages. Furthermore, based on the description of the most common types of AI, an assignment to the examples already given in Table 2 will be provided. ML and DL can broadly be clustered into Supervised and Unsupervised Learning and on whether the technique deals with discrete or continuous concepts. The resulting 2×2 matrix is shown in **Figure 10** with the main AI applications incorporated into the scheme with classification being a supervised discrete, regression a supervised continuous, clustering an unsupervised discrete and dimensionality reduction an unsupervised continuous AI task. Association/Pattern search is an unsupervised task, which can be discrete or continuous.

Figure 11 shows a taxonomy of accompanying techniques to support the main applications, namely multitask-, reinforcement-, ensemble methods-, and DL techniques. Here, supervised and unsupervised techniques, together with semisupervised techniques, are assigned as classical techniques.

Supervised Learning is the technique upon which AI applications rely in around 70% of the cases. In 10–20% of the cases, unsupervised, semisupervised, and Reinforcement Learning is also applied.[4,31] The main difference in Supervised Learning compared with Unsupervised Learning is that the output data is being given, mostly in the form of labeled data. Therefore, the desired output is known due to external assistance. After training, the algorithms detect trained patterns to predict the label on additional unlabeled input data. Assigned to this learning domain in AI are techniques like classification, regression, prediction, and gradient boosting.[4,31]

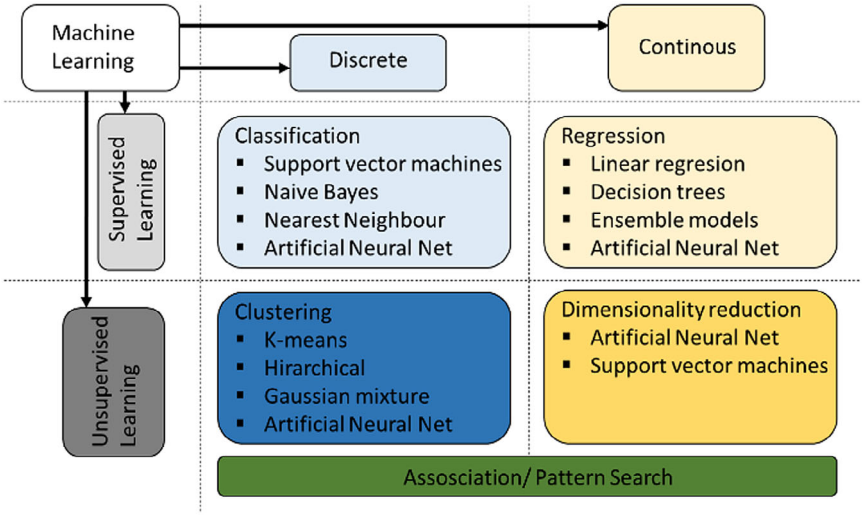


Figure 10. Main applications of AI discriminated according to defining attributes with exemplary algorithms.

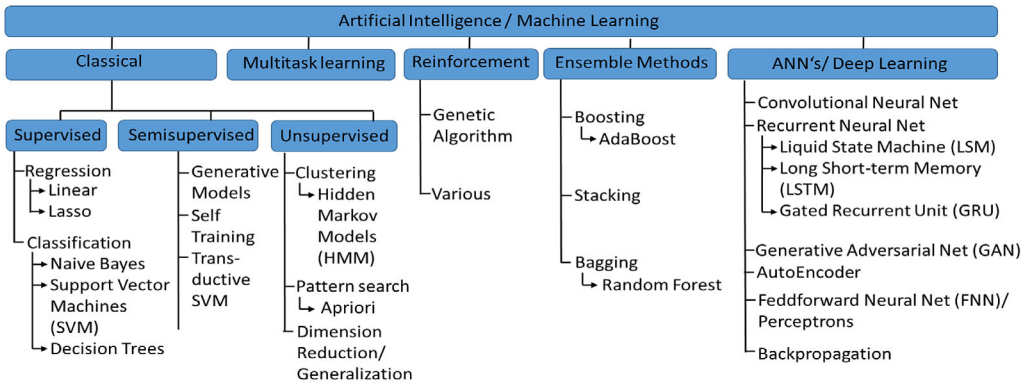


Figure 11. Taxonomy of AI-Classes and Types.

Unsupervised Learning, in contrast, differs in that the algorithms have no access to labeled data. Therefore, they have to explore the data by themselves to find inherent patterns and structures based on similar attributes. Among the algorithms are self-organizing maps, nearest-neighbor mapping, k-means clustering and singular value decomposition. It is predominantly used for clustering and feature reduction.[4,31]

Classification has the objective to allocate given objects to a discrete class or group, in which the classes and groups are defined and known beforehand. Therefore, the training is carried out on the basis of labeled data, which is already dedicated to one of the predefined classes or groups.[32]

Regression deals with making estimations or predictions for output variables along a continuum. Training input data are fed to the system alongside assigned output data, which is the known response of the respective input variables. This task encompasses the spectrum from simple statistical regression techniques to complex DL techniques for the purpose of making predictions within highly complex systems.[32]

Clustering sorts input data into discrete groups similar to classification. In contrast to Classification, the training data are not labeled, i.e., the output variables are unknown to the system. During training, the system learns to cluster the input data based on similar features and to make predictions on related features of new input data based on their association to a recognized group. The overall number of such groups can be prespecified or not.[32] The identified clusters may serve as input in the form of features to further AI-methods, or the user can assess the underlying meaning and consequences of the identified characteristics.

Dimension Reduction is used to compress a set of input data originating from a wide range of sources (e.g., sensors) with a consequently high degree of multidimensionality to a data set of reduced dimensionality. The resulting data can be used in subsequent data analysis.[32]

Semisupervised Learning represents an intermediary technique, which is similar to Supervised Learning, except it has access to labeled and unlabeled data at the same time. Usually, the set of unlabeled data is larger than the labeled set. Methods including classification, regression, and prediction are used to allow the AI to do its own labeling.[4]

Reinforcement Learning is a technique in which the algorithm (the agent) has the goal to maximize a reward based on the chosen action. Here, the agent receives feedback from the environment, and optimizes its actions toward the greatest reward by trial and error. An example of a reinforcement method is the use of a GA.[4] Therefore, Reinforcement Learning is a learning technique operating on the basis of a continuous inflow of new training data. This Online Learning approach differs from the Batch Learning approach, relying on a fixed set of training data collected before training. Historically, batch approaches used to be the dominant method of training AI's with online approaches becoming more prominent in recent years as Big Data acquisition provides a continuous inflow of data. Other recent approaches related to Online Learning are presented later, like ACRLs, Generative Adversarial Networks (GANs) or Active Learning.

Multitask Learning, also referred to as inductive transfer mechanism, deals with applying successful procedures from shared experience in past solution processes to different but similar AI problems and tasks, with the goal to facilitate and accelerate learning processes.[31]

Ensemble methods describe the combined usage of multiple learners to form a new single learning algorithm. In many cases, such ensembles of target-specific designs show higher performance than the usage of individual forms. Popular ensemble methods are **Boosting** to decrease bias and variance via the creation of a strong learner, or **Bagging** to increase accuracy and stability.[31]

ANNs and DNNs are the most prominent AI techniques covering all the aforementioned paradigms of learning. They are inspired by the way the human–brain processes information. The basic concept of a neural net is a neuron (also called node). As basic states, the neurons can either be active or inactive. Connections between the neurons exist, whose structure depends on the type of ANN. Generally, the net is divided in three basic types of layers, the input layer receiving the raw input data (pixels of pictures, numbers, etc.), a number of hidden layers, which are responsible for deriving the actual model, and the output layer, which display the output of the system. This structure is shown in **Figure 12**.

Each connection between 2 neurons has a specific weight. These weights are adjusted during the learning phase (training) of the network. During operation, signals are sent from the input neuron to the hidden layers and finally to the output layer, in a process called forward propagation. Within the hidden neurons the input signals are multiplied with the respective weights of the

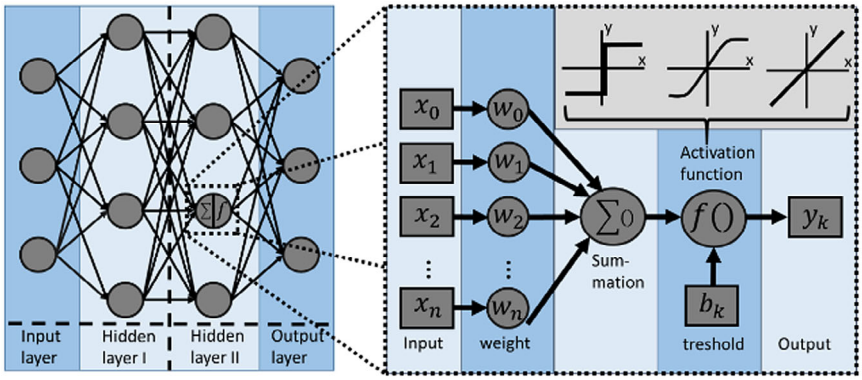


Figure 12. Basic functionality of a neural net.

connections, linking the respective neuron. These weighted signals are summed up in the neuron. Based on the combined signal, the activation function calculates the output. There are many variants of activation functions, many are continuous. The activation function resembling its biological inspiration is a discrete one, in which a specified threshold needs to be exceeded in order for the neuron to fire to the subsequent layer of hidden cells. More commonly, activation functions with less discrete transitions are chosen. After passing all hidden layers, the final layer represents the output layer. Training the weights (and optionally the thresholds) is carried out by reducing the error toward predicting the output in case of Supervised Learning. In BP, the weights are then adjusted based on the magnitude of the prediction error. This process is iteratively carried out, while utilizing the training data. In case of Unsupervised Learning, the network categorizes the data and evaluates the correlations without given correct output data corresponding to the input data.[6,31] By training, the network features are detected over the nodes in the hidden cells as reoccurring patterns form the input-cell level or lower-level hidden cells. Hidden layers closer to the output cells detect patterns of higher order and complexity on the basis of already learned patterns of lower order and complexity in a hierarchical way. Simple low-level patterns like geometric shapes, sound frequencies, and gradients in process conditions detected on low-level hidden layers act as input for higher level patterns like faces, sentences and complex plant models. Therefore, deeper neural nets enable the modeling of increasingly complex systems.

Table 3 shows an overview of the most relevant types of AI, mostly types of neural nets, with a depiction and a short description together with the major advantages and disadvantages.

Complementing to the already listed examples in Table 2, **Table 4** shows information for the listed examples regarding the performed AI-specific task (Classification, Prediction, Clustering, Association, and Dimensionality Reduction). If applied, also the method of preprocessing, the respective used AI methods, the required amount of training and testing data as well as additional relevant information, like the integration as a hybrid model or the use of Reinforcement Learning, are given. In some cases, multiple AI methods were used in order of comparing their performance for the given usecase. The IDs of the examples act as assignment to the lines in Table 2 to link the respective process engineering task.

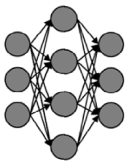
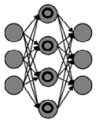
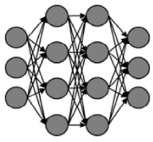
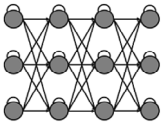
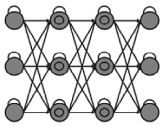
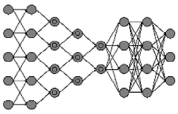
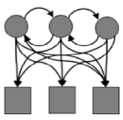
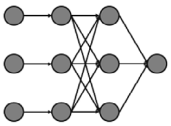
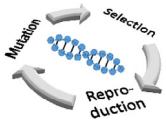
6. Roadmap to AI Modeling in Process Engineering

This chapter has the objective to investigate the overall workflow when applying AI methods to applications in process engineering as well as not-topic-related application cases in general. The steps described in this chapter accompany the actual core-AI-applications in process engineering discussed in Section 4, namely process predictive modeling, process optimization, process control, and fault/anomaly detection. Each of these steps yields great importance for the overall success of the AI training. Without proper consideration of all the factors accompanying actual training, the successful application of AI to achieve a working and evaluated model may not be feasible or lead to inappropriate models and conclusions. First, steps preceding the actual AI training are to be explored, which deal with the data foundation used for the actual training. This also encompasses data generation, as well as preprocessing strategies and the final evaluation of the AI with separate testing data, which was not used for training. Following this, the application and advantages of hybrid models are examined. Finally, reverse engineering strategies are examined with the objective of breaking down the trained empirical black box or hybrid gray box models to mechanistic white box models. The terms are going to be explained in this context.

6.1. Data Foundation, Data Generation, Preprocessing, and Limited Data Training Strategies

As previously discussed one important factor for the rapid growth of DL in recent years has been the advent of Big Data, the exponential growth of data due to the improving price performance and availability of sensors and the increasing interconnectedness of sensors, machines, plants, data banks, and humans, sometimes referred to as the "Internet of Things" (IoT) or the "Internet of Everything" (IoE). According to a Seagate-sponsored whitepaper, the worlds' data are doubling every 2 years, expecting to reach 175 zettabytes by 2025.[33] However, this extensive growth does not apply to all sectors i.e., industries, plants, machines, and processes to the same extent. Especially, on scientific level, data acquisition mostly relies on gaining experimental data through measurements,

Table 3. Often used types of ML algorithms with the most distinctive features and advantages and disadvantages.

AI-Type	Draft	Explanation	Advantages	Disadvantages
Simple feedforward Neural Net—Perceptron[93]		One directional Simplest form of neural net Perceptron: Single layer ANN Multilayer ANN	Fast to train Limited layers/ complexity Little computing effort in comparison to DL	Limitations in complexity of recognized patterns Requires more data without BP
ANN with BP		Cycle in nodes Weight adjustment with mean squared error	Higher accuracy Computationally cheaper Higher generalizability	Sensitive to noisy data Gradient decent Cost function derivatives
DNN[94,95]		Many hidden layers Gradient decent problem solved	Complex high-level/ dim. Features Robust scalable Generalization	Not suited for sequential data Big data/computing Overfitting
Recurrent Neural Net[96]		Output as feedback for input to retain memory	Capture of temporal or sequential patterns is possible	Vanishing/Exploding gradient Unstable Long distance
Long Short-Term Memory[97]		More complex neurons with additional forget gate	Solves vanishing and exploding gradient problem Long distance	Long training duration More RAM required Overfitting
Convolutional Neural Net[98]		Convolutional and pooling layers Feature dimensions are reduced	More abstract representations Fewer parameters Good weight optimization	No spatial, only statistical consideration Slow/demanding Hyperparameters Big data
(Hierarchical) Hidden Markov Model[99]		Based on Markov chains Hierarchical variant	Sequential data Long distance patterns with HHMM's	No Long distance patterns with HMM's Expensive
Support Vector Machine[100]		Hyperplane creation for clustering One output	Combinations with ANN's No local minima No overfitting High dim. data	Can become very big, but limited i n data amount Only one output Sensitive for noisy data
GA		Iterative adjustment of parameter sets emulating evolution	Search through big data sets Alternative to Reinforcement Learning	Limited to optimization problems

either online or offline. Likewise, process development is often based on a very limited amount of data extracted from pilot plant operation. This results in a very limited amount of data, especially when compared with Big Data scale. Therefore, it is vital to carefully consider the data origin and generation strategy for one's respective projects, with usually the main source of data

Table 4. AI specific details of the process engineering examples featured in Table 2.

	ID	AI-task	Preprocessing	AI-Tools		Information
[36]	3.a	**Regression**	Random sampling (DoE), LHS (DoE) and QBS (AL)	Gradient Boosting (XGBoost and LGBM)	2 sets with 8192 separate observations each	
	3.b	**Regression**	Random sampling (DoE), LHS (DoE) and QBS (AL)	Gradient Boosting (XGBoost and LGBM)	36 475 separate sensor measurements	
[26]	7.a	**Regression**	—	FNN (2 layers)		Hybrid
	7.b	**Regression**	—	FNN (2 layers)	1500 training datasets	Hybrid
[25]	1.1	**Dimensionality reduction**	—	ANN (AE)	259 PTA production samples (172 training, 87 testing)	Unsupervised
	1.2	**Regression**	—	ANN (Extreme Learning machine)		Supervised
[29]	8.a	**Classification**	Some preprocessing	DL (fully connected); Linear classical methods; SVM ; Random forest; DT; RNN; Adaptive Boosting; Gradient Boosting; Lasso	5 250 000 records	
	8.b	**Classification**	x (for LSTM and GRU)	DL; Linear classical methods; LSTM; GRU; SVM; Random forest; DT; RNN; Adaptive Boosting; Gradient Boosting; Lasso	274 628 records	
[90]	4	**Classification**	Gray-Level Co-occurrence matrix)	ANN; SVM; Random forest; DT (J48)	60 images	Supervised
[91]	5	**Regression**	—	DL	630 000 (training) 157 000 (testing)	
[92]	6	**Clustering** (1st) **Classification** (2nd) **Regression** (2nd)	—	SVM		Unsupervised (1st) Supervised(2nd)
[27]	2.1	**Dimensionality reduction**	—	DL (deep AE); SVM	16 000 laser welding images	Unsupervised
	2.2	**Regression**	—	DL		Supervised Reinforcement (nexting)
	2.3	**Regression**	—	DL	Simulation: 30 epochs	Reinforcement (actor-critique)
[30]	9	**Classification**	—	BP ANN AE-Softmax LSTM	1000 feature vector data set (900 training; 100 testing)	Supervised

being databanks, online measurements of existing distributed sensors, computer simulations, and experiments. Regarding the necessary amount of the sample size required for successful AI training, there exists no rule of thumb to anticipate the required amount. The necessary data depend on many factors, like the complexity of the underlying problem, which is often unknown, and the complexity of the used type of AI, the dimensionality of the input and output parameters, the number of hidden nodes and layers, the type of network, etc. (details can be found in Sections 5 and 6). In addition, the quality of the data or the amount of noise has a big impact. It is worth to mention, that there exist methods like data augmentation or feature extraction which have the potential to reduce the required data (see Section 6).

For the application of AI to a process engineering problem, a literature survey should be performed beforehand with the focus on previously addressed cases with a similar problem structure, within or outside the targeted field. A preliminary transfer of the given problem to a higher abstract level is advisable. After identifying a set of similar cases, the relevant information is to be compared, e.g., the performed pre- and postprocessing on the basis of the datasets as well as their respective size and format. Also, the used AI techniques and algorithms might have a high probability of being applicable to one's respective usecase. Additional relevant information might be the chosen hyperparameters, evaluation strategies of the trained nets, challenges that arose during the literature projects and discussed dis-/ advantages of the described setups.

While searching for similar literature examples but also for the later application of AI training, the overall problem should be broken down to smaller subproblems. Therefore, the complexity of the final system as well as the amount of required training data can be reduced. For example, regarding a chemical or any other process, this could be translated into dividing the process or the examined plant into submodules or aggregates, as is done in flow-chart simulations. The dedicated predictive AIs can then be trained in parallel and can be interconnected at a later stage.

For the overall problem case, as for divided subproblems, it is also important to carefully state the key problem (e.g., according to which criteria a process optimization should optimize the process), the context and the objectives. This was also suggested by Sokolov et al. who build a fault/anomaly predictor for a gas pipeline, leading to difficulties in detecting the normal state of operation due to the complexity of the problem.[29]

Depending on the source of data, the data should be formatted in a way that it is best suited for the respective chosen AI algorithm. Usually, it is important to bring all instances in your data set into the exact same format, so that for instance an ANN can find repeating structures in the data. Otherwise, inconsistent data formats will add additional complexity to the problem. Depending on the usecase, it may be required to filter, modify, and clean the data before training. **Data augmentation** is the modification of existing data, for example, by zooming in, reflection, shearing, rotating images, or by showing 3D images from different angles without changing underlying information and is a potential way to compensate for limited training data.[34]

Biased training data originating from data generation or induced by human labeling or processing should be avoided. Furthermore, the data should be divided randomly into a data set for AI training, and a dataset to evaluate it afterwards. Evaluation with training data instead of with dedicated testing data, may otherwise lead to wrongly correct results by comparing overfitted data with the very data used to train the overfitted algorithm in the first place. In the examined literature, the ratio of training and testing data most often found was 70:30, this may however differ depending on the usecase.

In addition to filtering, cleaning, preselecting, and debiasing the training data, the data can also be prepared for the training by emphasizing the main features in a process called **Feature Extraction**. As aforementioned in Section 4, the unsupervised technique of Feature Extraction was applied using an AE, a special form of neural net with BP, shown in **Figure 13**.[25,27] The main benefit of AEs lies in their ability to break down complex raw date into a representation by reducing the data to main features without noise and redundancies. This can be helpful for subsequent ML training, as it reduces the complexity of the input data and, therefore, facilitates the ability for pattern recognition. AEs are constructed in a way, that the number of input neurons is always the same as the number of output neurons, with at least one hidden layer with fewer numbers of hidden neurons. The input side is called encoder, the output side decoder. As the hidden layers are smaller, it is required for the encoder to reduce

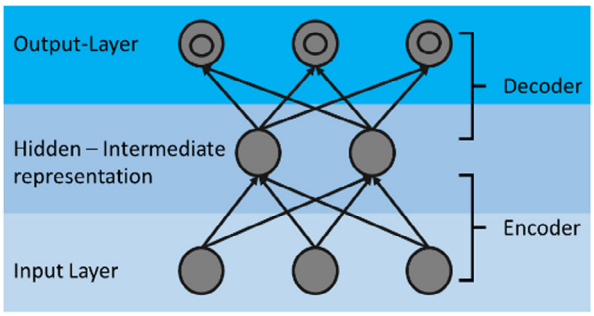


Figure 13. Architecture of an AE.

and simplify the incoming data to their main features, and therefore to learn a concentrated abstraction of the more complex raw data. It is the task of the decoder to reconstruct the original input data.[25,35]

AEs are typically used in **Dimensionality Reduction**. Reducing the number of dimensions used for training the network before the actual training process decreases the complexity of the data set and facilitates the actual training process. In addition, when investigating the input and output parameters, a sensitivity analysis should be carried out beforehand to exclude parameters without noteworthy impact on the overall system. This reduces dimensionality, which is beneficial, as higher input dimensionality reduces the training effect of neural nets. For instance, Xingyu et al. [30] reduced the number of indicators from 30 to 24 before training a fault detection system of a grinding process.[30]

As mentioned, the data source for AI applications could be experiments, simulations databanks, or other means. In case of databanks, the information flow is usually one directional, with the data being used to train and test the AI system. However, in case of experiments and especially simulations, the flow of information could also be bidirectional. One possibility for reducing the amount of required data for AI training represents predictive sampling, also called **Active Learning** (AL). The core concept is to let the ML algorithm decide which sample points to choose to train the model with minimal data. In the method two components exist, the query algorithm and the learning algorithm. Here, the sample selection is done by the AI algorithm within the unlabeled training data, the sampling (within databanks, or by carrying out simulations or experiments) is then executed and labeled by the so called oracle for continued training of the ML algorithm. Samsonov et al.[36] used such a method for building a predictive model for the dynamic behavior of a robotic arm and the temperature behavior of an electric motor and compared the performance of the AE strategy with a classical Design of Experiment (DoE) strategy.[36]

While training the network with limited data, one of the biggest risks lies in overfitting, which refers to an overly high degree of coadaptation of the network to the training data. The resulting network is unable to generalize and overrates the noise in the training data. Different countermeasures have been proposed, like **stopping training** after detecting a worsening in prediction accuracy during the validation of the trained network or the **introduction of weight penalties, a method to keep weights small, and therefore to prevent overfitting**. One approach to encounter that problem is the application of a technique called **Dropout**. Here, neurons and their respective connections in the net are randomly deleted, preventing excessive coadaptation to the training data.[37]

A good way to generate data and to train a neural net (or other AI technique) is to set up cases in a way that either automated quick experiments are the foundation, or a quick to run simulation setup, e.g., a population balance setup of a process can be used for **(Deep) Reinforcement Learning**. Here, the learning algorithm usually starts without any previous data, the learning process is carried out dynamically as the algorithm receives constant feedback from the simulative (or experimental) outcome of its given output predictions, acting as input parameters for the simulation. In Deep Reinforcement Learning, an AI algorithm (agent) is given the task to reach a goal, like controlling or optimizing a process, and promotes this via trial and error, whereas

its predictive ability increases over time. A popular example for this method was applied at the famous Alpha Go challenge in 2016. An AI beat the best Go player in the world after being given a simulation containing the rules of Go. The algorithm iteratively became better and surpassed the best alternative softwares and human players after a few hundred thousands of training steps against itself.[38]

In the case of Go, a simulation based on the exact rules of a system was used, which extends to simulations on the basis of physical or established mechanistic models in the case of process engineering (white box models, see Section 6.2). Alternatively, simulation environments can be built either on purely empirical models or by combining known laws with empirical data. For instance, Waymo, the autonomous driving Alphabet/GOOGLE subsidiary company, is integrating real-world data from autonomous cars to feed them into an empirical model "Carcraft." The simulated cars drive significantly more miles (billions) than the real cars on actual physical roads (millions), accelerating the training process.[39] Therefore, this method can also be applied for actual experimental setups by dividing the training effort into a limited number of real-world experiments and a higher number of simulated experiments, based on data of real-world examples.[40] Within the simulation the data can be modified to generate a larger set of "situations." The application of ACRL is an additional technique, which can be used in conjunction with the aforementioned examples of coupling with simulations and Reinforcement Learning. In ACRL, the actors' task is to learn a policy to generate actions. Its performance is evaluated by the critic. The process is run sequentially, resulting in an increasing performance in prediction making. For instance, in mechanical engineering, ACRL was used to establish a process control for a laser welding process. Preceding steps for the evaluation of laser welding performance were already discussed in Section 4. Process control, building on top of these, could respond to the chosen actions of the actor by applying them to the results of a laser welding simulation, running the actor's parameter choices.[27]

A technique with some similarities to ACRL are **Generative Adversarial Networks** (GANs), which are applied in Unsupervised Learning. Here, the opposing networks are called generator and discriminator. The generator is tasked with generating candidates, resembling an expected result, with the discriminator deciding whether the candidates were generated or not. In the process, the two networks train each other. The discriminator trains the generator by evaluating its generated outcome and the generator trains the discriminator by generating increasingly convincing data. Only the discriminator has access to original data, the generator is supervised by the discriminators supervision alone.[41] A popular application is the generation of artificially constructed or modified picture and video material, which has led to the emerges of the infamous "deep fakes."[42] The main differences to ARCLs are the usage of GANs in Unsupervised Learning instead of Reinforcement Learning. GANs main directive is the generation of convincing data distributions to be indistinguishable from original distributions, whereas ACRL directive is the investigation of given environments and the according optimization of its actions in regard to it with specified goals.[41] A potential application is the use in anomaly detection[43] or data augmentation.[44]

All previously mentioned techniques and preliminary or preprocessing steps are potential aids when facing AI problems with limited data. When possible, it is usually suggested to increase the underlying database. However, this is not always possible due to logistical or financial reasons. When beginning the actual training starting with simpler algorithms like regression algorithm, random forest, or genetic codes is a good option to gain first insights. In addition, it allows the estimation of the expense and progress, also by tracking the training performance in dependence of the data amount and the chosen AI method.

One possibility to strongly reduce the amount of required data and training duration is to perform so called **Transfer Learning** on the basis of **pretrained models.** In Transfer Learning, data sets of similar usecases are used to train a model, resulting in a model in which many similar features of the final desired usecase are already present. Instead of starting with random weights in the network, this pretrained model can then be trained on the basis of the data corresponding to the actual desired usecase. With this knowledge transfer, a ML model can be trained on a relatively small high-quality data set. The only prerequisites a bigger data set of a related task domain, in which the task as well as feature space and distribution may vary. Generally, in the case of closer related task domains, Transfer Learning is more successful. In the case of completely unrelated source and target domains, Transfer Learning may however even weaken performance, called negative transfer. Depending on whether labeled data in source or target domain is available, different types of Transfer Learning are possible. They are referred to as transductive (labeled source), inductive (labeled target), and unsupervised (unlabeled) Transfer Learning, with the first two being used more often. Moreover, **Semisupervised Learning** is a promising option also in regard of Transfer Learning.[45] An example in process engineering was described by Yuan et al., combining semisupervised stacked AEs for pretraining a network with fine-tuning on the usecases of a debutanizer column and a hydrocracking process.[46] Semisupervised Learning represents a convenient technique in cases were input data are abundant, but obtaining larger amounts of respective labels is deemed as too expensive or impossible on a larger scale. The technique is usually chosen for classification tasks, often in combination with generative models. Abiodun et al. demonstrated its superior performance in contrast to Supervised Learning with limited data by applying this method on the basis of a combination, consisting of probabilistic modeling and DL.[47]

Concerning the use of DL **in cases with limited data** sets, a common assumption is that it will cause an unavoidable overfitting of the data. However, when implemented correctly, also DNNs are capable of sufficient generalization in such cases. Olson et al.[48] demonstrated an approach contrasting the conventional approach of finding an architecture suitable of fitting a given data set and subsequently scaling back through regularization. In Olsen's method, an approach similar to random forest (using deep DTs to highly accurately fit the data, with subsequent randomization and averaging to reduce variance) can be applied to DL, suggesting the final layers of the DL to act as an ensemble method. An alternative strategy to prevent overfitting lies in applying methods of redundancy reduction, by pruning weights without significant effects (a technique similar to Dropout). Methods to accomplish this can be penalizing algorithms to reduce network complexity by performing sensitivity analysis of prediction error to the weights of the connections between

the neurons, with the goal of eliminating connections with little effect on the overall fit of the trained model. In addition, during the training of network weights, these can be frozen in case of deteriorating prediction accuracy upon training intervals.[49]

After the **training**, the network needs to be evaluated. **Evaluation** is performed with independent testing data not used in the training phase to prevent false-positive results in case of overfitted networks. Usually, the data are divided in training and testing data, with ratios usually varying between 70% and 90% training and 10–30% testing data. More precise evaluation methods include the discrimination between true positive or negative rates, confusion matrixes or the application of Receiver Operating Characteristic (ROC) curves for visualization. An overview is provided by Kiranmai and Damodaram.[50] To emphasize the differentiation of terms, Evaluation is the application of the trained AI to test its accuracy and robustness. The regular application of the trained and evaluated AI to its later usecase is called **Inference**.

6.2. Mechanistic Model, Transparency, and Hybrid Models

The previously discussed methods and algorithms focus on the sole training of ML algorithms as black boxes, with little to no transparency about their underlying trained and recognized patterns, which were obtained during training on the basis of empirical data. As discussed in the following, methods exist to investigate these patterns to gain mechanistic insight and to derive mechanistic models. Mechanistic models as well as physical models are called white box models, characterizing models whose internal mechanisms and correlations are perfectly known. A promising intermediary form is the use of gray box models, also known as hybrid models. In case of hybrid models prior first principle "white box" models need to exist. These act as preliminary model to provide initial correlations, especially in the case of only sparse and noisy data. The ML "black box" model supplements the first-principle preliminary model to capture unknown functional relationships in the investigated system to achieve accurate, consistent, and reliable predictions.[51] A broad continuum exists, concerning the extent to which these hybrid models contain known white box models and trained black box models. In addition, there is a large variety of possibilities in the way these can be interconnected. For instance, based on whether the experimental results are applied on the basis of a theoretical or a quantitative model, gray box models can be subdivided in light gray and dark gray models, as shown in **Figure 14**.

Hybrid models have a multitude of advantages in relation to the sole usage of white box or black box models. They require significantly less data in comparison with the training of pure black box models, due to the existing predetermined structures, which reduce the dimensionality of the data and facilitate the

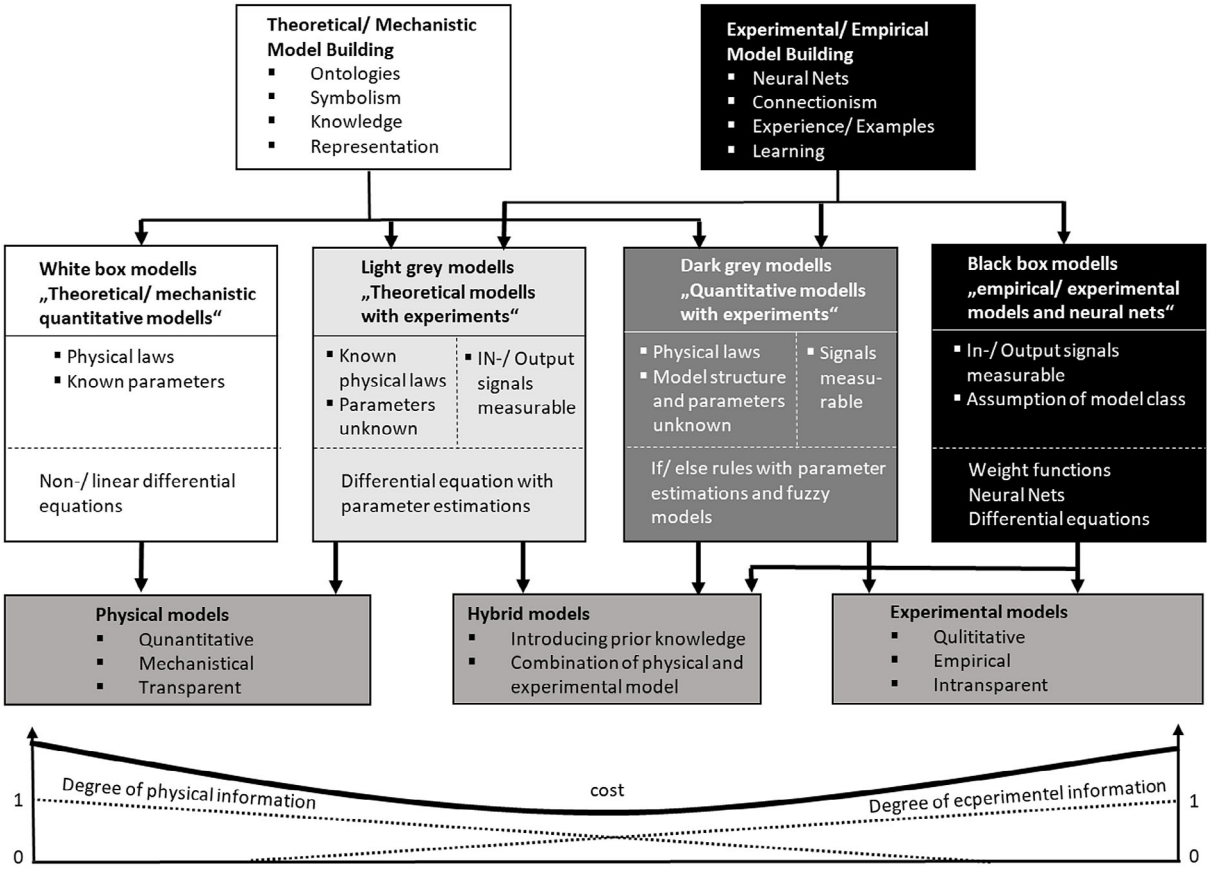


Figure 14. Overview of the continuum of white box and black box models with their dedicated tasks, input source, types of mathematical representations, and resulting types of models.[89]

pattern recognition for ML algorithms like neural nets. Therefore, training can be reduced. Usually, more accurate and consistent models can be derived and extrapolations be made. Due to their semitransparent nature and the reduced complexity resulting from the more modular structure, a reverse engineering is also easier to achieve, see Section 6.3 for details. The smaller size of the more modular ML-models is also a reason for the lower required training effort due to the underlying breaking down of the system into easier to model subsystems. As a result, the usage of hybrid models usually reaches a better ratio of afford and resulting performance.

Hybrid models can be constructed in multiple ways. Often modular structures or semiparametric structures are used, as shown in **Figure 15**. In case of modular structures, multiple interconnected ML algorithms (e.g., neural nets) are set up to each represent and model a subsystem in a broader system on the basis of prior knowledge about the assembly of the investigated system. Therefore, the topological and functional structure of the submodules resembles the structure of the system under examination. For instance, in case of the predictive modeling of plant behavior, instead of building one big neural net, subnets could be established to each represent an aggregate with the input and output nodes representing the relevant parameters through which the aggregates influence each other (e.g. pressure, temperature, and so on). In case of a semiparametric model, an AI algorithm is used in tandem with a first-principles model, which sets the overall structure. A semiparametric model can be set up in a serial manner with the first-principle model giving a broad estimate of an output and the ML-model estimating intermediary variables for the first-principle model. Alternatively, the first-principle model and the ML-model can be set up in parallel as their respective outputs are combined to the final output. With the ML-algorithm being trained on the residual between the data and the first-principle model, it can compensate its shortcomings and, therefore, can fine tune the overall model.[51] In many cases, combinations between these methods are used. In the field of process engineering, a multitude of AI applications were used in combination with hybrid models, usually performing with a higher efficiency than with the use of standalone networks.[52] A study of the use of hybrid models was carried out by Zendehboudi et al.[52] for different examples in chemical, petroleum, and energy systems. In the following, representative examples for every method are presented.

An example for a serial semiparametric model was shown by Sundram et al. who demonstrated a hybrid model for the prediction of the intake valve deposit formation in a motor. In this case, the first-principle models in the form of various existing models describing many involved phenomena preceded the neural net. The first-principle models alone were insufficient to build a purely first-principle model, but captured as much knowledge about the given system as possible. The subsequent neural net bridged the gap to the physical data, demonstrating the profound benefits of incorporating a priori knowledge to the best degree possible in the form of first-principle models and to use data-driven models to gain a more holistic system model on the basis of substantially less data, than would have been required in a standalone ML application.[53]

A parallel semiparametric hybrid approach for the behavior prediction of a biotechnological process with the aim of producing second-order biofuels from waste biomass was proposed by Curcio et al.[54] More specifically, the enzymatic transesterification of olive oil glycerides was modeled. The existing white box model covered kinetic rates, which were previously established and described the relationship between reactant and product concentrations. To compensate the still improper theoretical model, which potentially causes incorrect predictions of bioanalytical reactions and resulting temporal concentration developments, a neural net was constructed, as shown in **Figure 16**.

To more accurately model the substrate–product concentration relationship, the ANN, a pyramidal multilayer feedforward perceptron, was trained to predict the temporal ethyl oleate concentration difference on the basis of various input parameters such as different component mass and molar ratios, initial water content, reactor agitation rate, and substance consumption, etc. The resulting ethyl oleate formation model was used as an additional model running parallel to the existing model and feeding into it to consider intermediary not yet covered values.[54]

Thompsen et al. presented a hybrid model in the field of chemical engineering for modeling a biochemical process in the form of a penicillin fermentation. Two first-principle models were incorporated. The first being chemical balance equations describing cell biomass-, substrate-, product-, and overall mass balance. The second model was a set of equations describing the specific rate correlations like growth, cell lysis-, substrate consumption-, product formation rate, and maintenance energy.

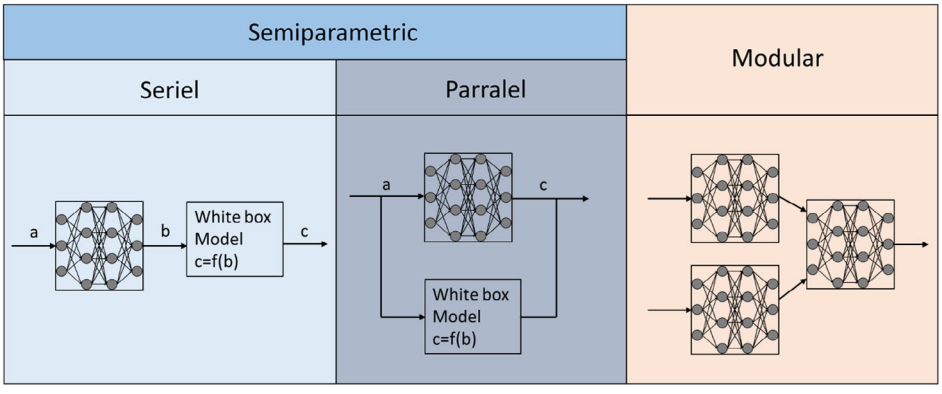


Figure 15. The main types of hybrid models of semiparametric and modular design.

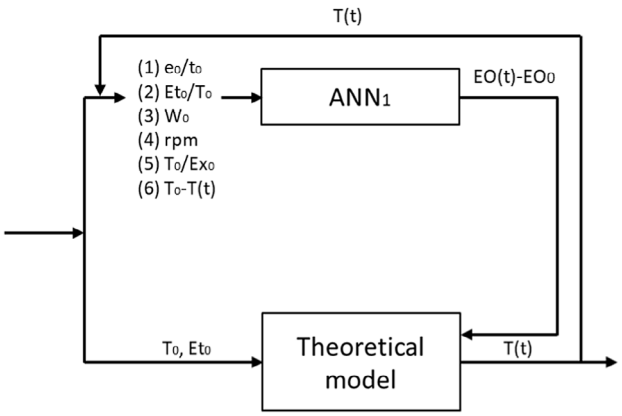


Figure 16. Schematic of the parallel model with the parallel ANN model describing the difficult to model ethyl oleate formation in dependence of various biocatalytical parameters as input parameter for the kinetic white box model. Reproduced under the terms of the CC BY 3.0 licence.[54] Copyright 2014, Hindawi.

Pre-existing knowledge from experiments was already included in the underlying model equations. On this basis, a parallel semi-parametric model was constructed. It included a radial basis function network (RBFN), a form of ANN, which is trained to compensate for the error in predicting the specific rates caused by hidden time-varying parameters of the described first-principle model. The corrected rates could then be used in the subsequent mass balance equations. The hybrid approach showed better performance in comparison with a standard BPN and a pure RBFN trained on the same training data without the hybrid modeling approach.[51]

Similarly, Tie et al. demonstrated an online soft sensor for a hydrocyclone overflow for the estimation of particle size distributions, containing multiple aggregates, shown in **Figure 17**.[55]

The approach followed a modular approach as the process was divided into its basic underlying functional structures. The hydro cyclone and the classifier are completely described via an ANN which was trained to predict the mass flow $\dot{M}_{75}$ of solids with a diameter under 75 μm as a function of its underlying parameters (masses, concentrations and respective flows) for the cyclone. For the classifier, the mass percentage M_{75} was predicted as a function of its input parameters like shaft power and recycle slurry flow rate. The sump was modeled with a first-principle population balance equation model. For the two neural nets for the cyclone and the classifier radial basis function (RBF) networks were chosen. Together, these three models were used to describe normal process conditions. For the consideration of abnormal process behavior, an ANFIS model was trained mainly for the prediction of malfunction of measurement meters or excessive slurries, etc. A subsequent fuzzy logic coordinator was used to combine the outputs of the two underlying models to one output, as shown in **Figure 18**.[55]

As mentioned earlier, combinations between such types of hybrid models are possible and often carried out. For instance, a hybrid approach combining a neural network with first-principle models was introduced for a fed batch bioreactor by Psichogios et al.[49] Existing kinematic equations served as first-principle model. In their scope, the parameter of cell growth rate has a big impact. However, cell growth rate is difficult to

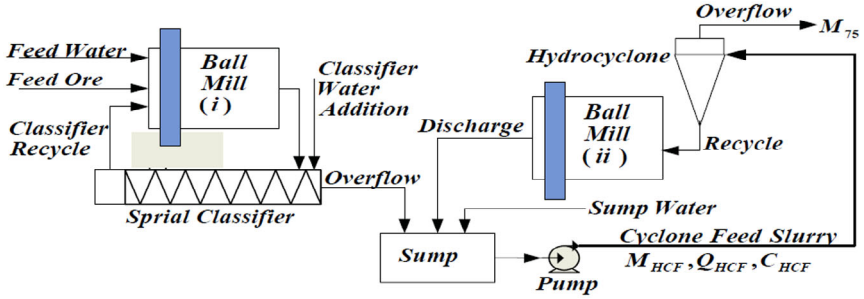


Figure 17. Schematic of plant comprising various aggregates for an ML estimation of particle size distribution via a soft sensor approach. Reproduced with permission.[55] Copyright 2005, Springer.

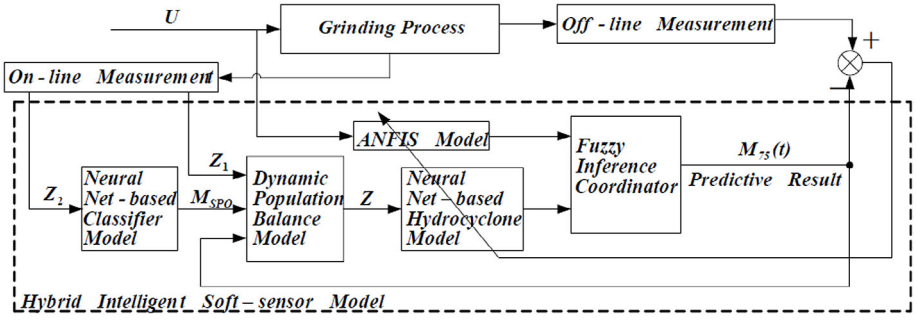


Figure 18. Modular approach for the modeling of a complex grinding plant for particle size distribution prediction. Reproduced with permission.[55] Copyright 2005, Springer.

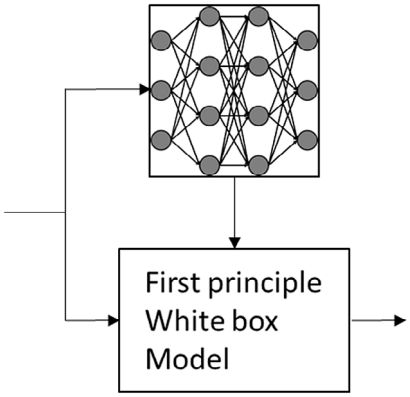


Figure 19. Combination of serial and parallel hybrid model methods into one architecture.[49]

model with a first-principle approach predestining it to be incorporated via hybrid modeling. The structure is shown in **Figure 19**.[49]

The incoming parameters like biomass, substrate, and substrate feed variables were fed to both the neural network and the first-principle models. The neural network used these variables as input parameters and was trained via BP to predict the cell growth rate. As no measured data for cell growth was available as output data for training, the partial process model was used to calculate an error signal, which could then be used to update the neural network weights. The predicted growth rates were subsequently passed to the first-principle kinematic equation models.[49] The methodologies were combined by breaking down the system to modules containing first-principle and neural net components. The neural net was used to precondition the data before entering the first-principle models in a serial way with the first-principle and the neural net component simultaneously running in parallel. Bhutani et al. investigated and compared the application of a serial, a parallel, and a combined serial–parallel hybrid model approach for an industrial hydrocracking unit, as shown in **Figure 20**.[56]

The plant consists of a complex wiring of different units, including furnaces, a hydrotreater, a hydrocracker, a high-pressure separator, a recycle gas compressor, a low-pressure separator, as well as multiple separation columns. The plant contains a multitude of components with heavy vacuum gas oil and makeup hydrogen as input components. Components like liquefied petroleum gas, light and heavy naphta, kerosene, light and heavy diesel, unconverted oil, off-gas, ammonia and hydrogen sulfide are circulating and leaving the plant. Two first-principle models capture aspects of the process but only achieve limited accuracy. In the study, a pure data-driven model (a pure ANN model) as well as setups of hybrid models were trained, one serial model, one parallel model, as well as a more complex combined serial–parallel model, as shown in **Figure 21**. The serial model describes uncertain parameters and time varying kinetic parameters. The parallel model describes the system behavior with the ANN and corrects the outputs. The combined model corrects the previously trained serial model via a parallel running ANN.[56]

All models were trained with data from 110 days of operation and were given the task to predict the plants' behavior for a time

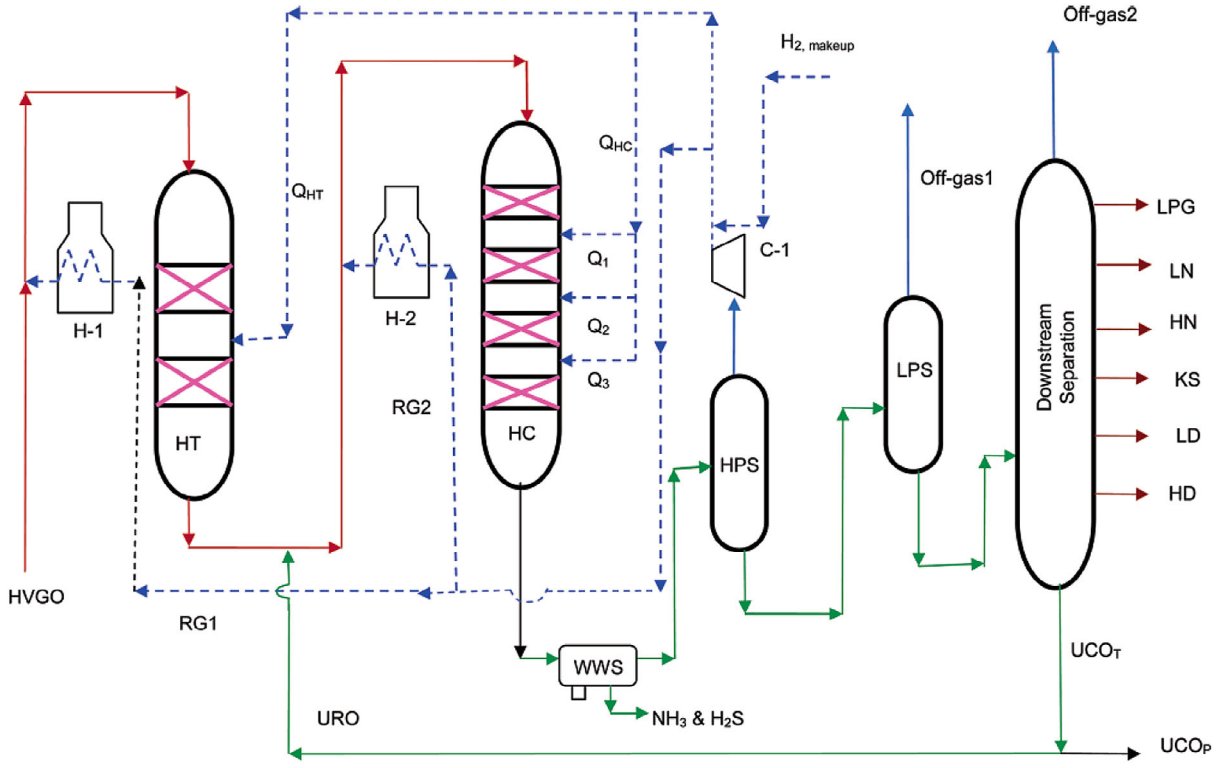


Figure 20. Process diagram of hydrocracking unit. Reproduced with permission.[56] Copyright 2006, ACS Publications.

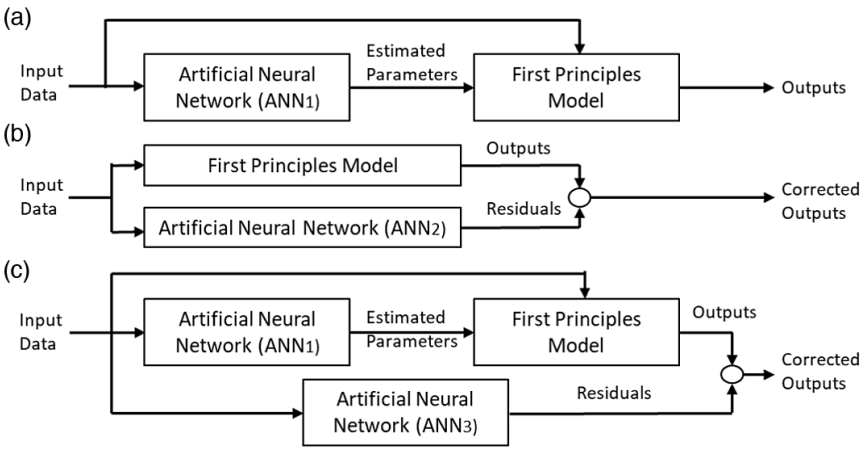


Figure 21. Different hybrid modeling approaches, a) serial, b) parallel, and c) serial–parallel. Reproduced with permission.[56] Copyright 2006, ACS Publications.

span of the following 5 days. A set of 18 input variables like feed flow rates, fractions, flows, temperatures, boiling points, and densities, was given to model output flow rates Feedforward multilayer neural nets with two hidden layers were used with 18 input variables in all cases. The number of output variables varied with 5 for the serial model and 13 output variables for the other neural nets, respectively. The parallel model proved to be easier to develop and achieved better accuracy then the serial model. The combined serial–parallel model was the most difficult to train. The pure data-driven model without a hybrid portion was easier to train in comparison, and may show better results than the parallel model.[56]

AI methods can also be used to develop mechanistic models for material properties or processes. Finke et al.[57] used a GA to identify a broadly mechanistic model for the viscosity of a nanoparticulate suspension. The model requires a detailed examination of surface interactions in the disperse system. While analytical solutions exist for the interaction between particles of various sizes, an expression needed to be found which correctly weights the contribution of each interaction to the overall stress state in the suspension. The GA was presented multiple plausible options for such a weighting factor and was allowed to choose between those options. Each option was assigned a probability of execution in the genome of the GAs individuals. Those individuals were favored during the execution of the GA, who exhibited a high execution probability for the best weighting option. In **Figure 22**, a schematic depiction shows, how over the course of multiple generations one option dominates the population, which can be considered the best suitable option. The identified model allowed the description of the very process-relevant viscosity with respect to the particle-size particle content, temperature, and shear rate and even allowed an extrapolation beyond the calibrated parameter range.

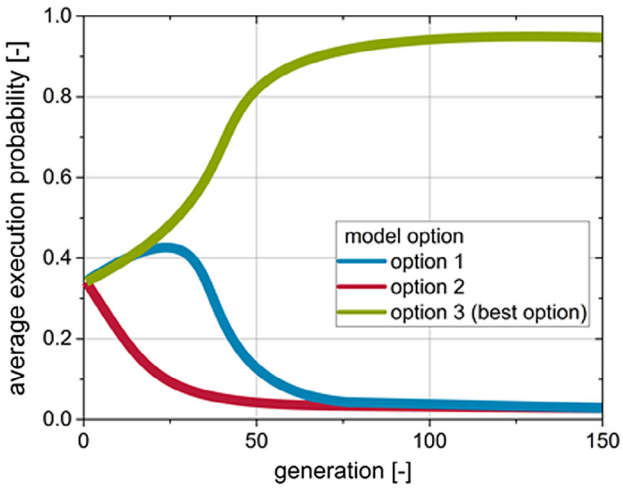


Figure 22. Development of the average execution probability of model options during a run of a GA. A large average execution probability translates into a dominance of the option in the GA's population.[57]

6.3. Reverse Engineering and Post Processing

In the previous chapters, the use and training of black box and gray box models have been discussed, which are both well suited for predictive modeling, and in the latter case also allow to gain some degree of mechanistic insights. In addition to the already useful predictive modeling, a main aim in science as well as in engineering is to set up mechanistic first-principle models, or white box models, which incorporate understood mechanisms of an investigated system. Gaining such models requires laborious effort and manual human involvement resulting in a time and cost-intensive endeavor. The use of computer simulations and automation in addition to the classical experimental approach has already helped reducing these expenditures; the use of ML in synergy with the mentioned advances represents the next step in further improving the modeling effort. Different AI techniques vary in their transparency, with the more performant techniques usually being also the least transparent. Therefore, the focus of this chapter is to present methods to break down the trained black and gray models in the pursuit of deriving mechanistic (white box/first principles) models.

Generally, two approaches exist to understand the trained and evaluated models, either by investigation of the formed

connections and behavior, by studying the structure as well as the performance of the network during operation, or using the trained network as a computationally inexpensive surrogate model for further analysis.

Studying the trained network to reverse engineer it and to gain insights is often summarized under the term "Explainable AI" or XAI. They serve as umbrella terms to summarize all methods that attempt to reveal the underlying patterns in a comprehensible way. As a first step, it is usually beneficial to reduce the complexity of trained networks before applying methods of reverse engineering. In the previous chapters, modular neural nets were already examined by assigning dedicated subtasks to dedicated subnets. Modular neural nets, as shown in **Figure 23**, can however also be applied as an architectural choice in designing the network, without prior knowledge. This approach is inspired by the neocortex also being divided into smaller subcomponents (so called cortical columns). Other AI techniques like Hierarchical Hidden Markov Models already incorporate such architectures inherently within their design. As most hierarchical AI techniques act as a hierarchy of detected patterns consisting of already recognized subpatterns of lower order and complexity, this modular approach can facilitate pattern recognition in contrast to training one big net. Due to the hierarchical nature and limited interconnectedness, reverse engineering of a modular architecture is also facilitated.[58]

An additional way for post-training complexity reduction is the already mentioned method of Dropout (see Section 6.1). In this case, Dropout is applied on individual neurons after training, not to prevent overfitting, but to study the effects on the overall network and the performance after the deletion of the respective neurons to evaluate its impact.[59] Applying this approach over all neurons of the net also enables to reduce the overall size of the network. The deletion of more important neurons will lead to higher deterioration of the accuracy with other deletions only leading to smaller deterioration or no damage at all. In the later cases, neurons can be deleted up to a point, which is still acceptable in terms of accuracy of the network. It holds the potential to reduce the network size in favor of network transparency and interpretability. Other methods like size penalties for too large networks may also be applied additionally to keep the network small. Often reoccurring methods of reverse engineering are described by Guidotti et al.[60] in which most of the following methods are described in greater detail. Only an exclusive selection of examples is displayed in this article.

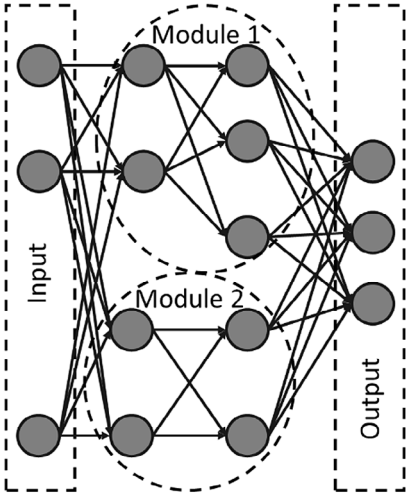


Figure 23. Schematic of a modular neural net.

Visualization encompasses a broad field of possibilities, many of the below mentioned methods also involve visualization to some degree or can be enhanced by its use. For instance, the activations on the different layers of the network can be studied for different inputs and outputs. The goal is to identify, which features are represented by the various hidden nodes and in which manner those features are derived from another. A hierarchical structure is found, with less complicated features being derived near the input layer and increasingly complex ones closer to the output layer. For different inputs, the change in the output can be studied. Furthermore, visualization of different nodes in hidden layers, especially in the field of picture recognition, allow the creation of activation atlases.[61]

Generalized Additive Modeling (GAM) is a technique used to construct an interpretable mathematical model on the basis of a trained otherwise in-transparent AI-algorithm like boosted trees, SVMs or DNNs. It uses a combination of various single-feature models (shape functions, describing the individual features) with linear models (the so called link function, which describes their correlation). These additive models are fitted with the ML algorithm, allowing to understand the contribution of individual features on the resulting prediction. The resulting function can be visualized to see the mathematical correlation of the individual features as a mathematical plot.[62]

Rationalization aims to deliver explanations for decisions made by ML systems in case of unexpected system behavior or failure in a human-like way by providing verbal or visual explanations. Ehsan et al. described the technique as a form of machine translation task, in which internal state-action representations are translated into a linguistic explanation.[63] This method requires human intervention to provide a training corpus by formulating verbal explanations for exemplary situations for given data sets. An alternative approach is "thinking aloud" while performing the respective task intended to be performed by the ML algorithm with their annotations, states and actions being fed to a separate ML algorithm learning to correlate them correctly.[63] A similar approach for verbal expression was demonstrated by Park et al. In this case, it was carried out in a multimodal methodology, in addition to giving visual explanations to the verbal explanation by focusing or pointing on objects or regions in pictures with the highest degree of relevance for the justification of the respective answers.[64] This approach provides a very human centric way of explanation. However, the necessity for human annotation presents a bottleneck, especially, as the given problem needs to be explainable for humans in the first place.

Gradient-based methods are used to highlight significant changes in units and features by applying gradients of BP, which can for instance be applied in the method of Prototype Selection (PS) mentioned in the following sections, while maximizing neuron activation.[61] A method known as Counterfactual method compares output layers by changing details of the input data after getting the result from the original input data.[65]

Local Interpretable Model-agnostic Explorations (LIME) works by breaking down input data into smaller and easier to interpret subparts. Concurrently, it shifts the demanding modeling from a global scale to a more apparent local scale. By evaluating the individual parts for significance, the most relevant features in the input data can be highlighted to receive an understandable explanation of how the network got to a respective prediction.[60]

Deconvolution is a method in which a trained ANN is applied inversely, generating the "typical" input for a given output.[66] The technique is often used in PS, described in the following sections. It can also be applied to generate typical inputs for specific hidden layers or nodes.[61]

Decomposition encompasses means to isolate, transfer, or limit fractions or layers of a trained network in regard to gain further insights studying the effects on the overall network.[61]

DTs comprise all techniques that make use of DTs as simpler and easier to understand concepts compared with neural nets. The first approach of this kind was presented by Crevan et al., who approximated and converted the underlying patterns into a DT by querying the network to mimic the behavior of the network.[67] It is often applied in conjunction with genetic programming, and can be applied to various ML technique like neural nets or random forest.

Decision Rules (DRs) describes techniques to break down a trained ML-algorithm into a set of rules, like if-then-rules. For instance, there are ways to break down rules from before mentioned DTs. Mechanisms exist for the insertion of rules into neural nets, their extraction as well as to refine existing rules. First shown by Craven et al.,[68] conjunctive DRs for a given network were derived whenever input–output-correlations were not covered in an existing rule set considering the overall space of possible antecedents.

Feature Importance (FI) is the technique of studying features together with their respective weights for given results.[60]

Salient Masks (SMs) display selective aspects or discriminative regions in data samples on which an ANN focuses during training and inference. By this, it identifies the origin of highest impact on the given predictions. For instance, on the basis of picture analysis, the parts of an image may be displayed, which include the main features detected by an ANN to identify a part of the overall pattern.[60]

Sensitivity Analysis (SA) applies for tabulator datasets (most datasets where features are numerical, categorical, or Boolean, with texts and images excluded). In SA, the uncertainties in correlated input and output data are compared and put into correlation. It can be evaluated to what extend input data or hidden features impact output predictions, which can act as a measure of transparency.[60] This technique can also be applied layerwise within the network via Layerwise Relevance Propagation (LRP) for the detection of intermediary features and their evaluation regarding their impact on the output of the network.[69] This method can also be used for the already discussed method of Dropout. The incorporation of visualization methods has been presented by Bien et al. who visualized explanation vectors highlighting features of high influence on the prediction result.[70]

Partial Dependence Plots (PDP) is a visualization technique which operates in a reduced feature space displaying the dependencies between input and output data.[60] Also independencies of features can be examined as demonstrated by Hooker et al.[71]

PS returns representative data for input data together with the respective outcome to illustrate the underlying features and criteria being characteristic for the given input data. Thus, through assigning outputs, instances or labels to representative prototypes, a higher degree of interpretability can be achieved.[60]

Neurons Activation (NA) describes the study of neuron activations for given inputs.[60] Interpretation can be further facilitated through the selective activation of neurons from layers of interest. Insight is gained, by studying the resulting activation schemes and outputs or inputs via the inverse application of the trained net.[61]

The discussed methods of studying the trained neutral nets (or other ML algorithms) to gain mechanistic insights are based solely on the existing trained nets. However, this approach is limited by the complexity of the investigated system and involves a high degree of human ingenuity and common sense. A more promising approach is to use the given trained model as a **surrogate model** for subsequent analysis algorithms to convert the neural net into mathematical functions containing the patterns recognized by the neural net. One of the most promising methods lies in the use of GAs, sometimes itself assigned as an AI or ML technique.

ML **accelerated Genetic Algorithm (MLaGA)** refers to the use of ML and a successive GA. Jennings et al.[72] give a good example for using a trained network as a computationally inexpensive surrogate model for further analysis. It is applied to the search of stable, compositionally variant nanoparticle alloys in the field of accelerated material discovery. In the publication, a 147-atom-structure chemical ordering process had to be optimized. Due to the high-dimensional feature space, a high number of potential permutations exist (146 compositions and over 10^{44} possible homotops), which cannot be tested with simulations. With the single application of a GA, the hull of local minima could be located, including the energy evaluations of only 16 000 homotops of the overall 10^{44} potential candidates. With the MLaGA approach, this could be further reduced to only around 300 homotops. The MLaGA approach combined the robustness of a GA with the rapid learning capabilities of ML. It resulted in a 50-fold reduction of the required computationally expensive energy calculations, compared with the sole brute-force application of a GA. GAs often require a high amount of function evaluations, which makes their use difficult, if the evaluation involves experiments or time consuming simulations.[72] A trained ML technique as surrogate model for evaluation instead of experiments or long simulations reduces the overall application time and enables the consideration of more data points leading to a more accurate modeling of the GA. The flowchart describing the modeling process with the MLaGA is shown in **Figure 24**.

On the basis of an initial population of 150 candidates, classical fitness evaluations (demanding simulations for the calculation of the homotope excess energy) were carried out to be used as training data for the ML approach, a Gaussian Process (GP) regression model. Instead of GP any other ML-model (like DL) could be applied. Two types of GAs were then

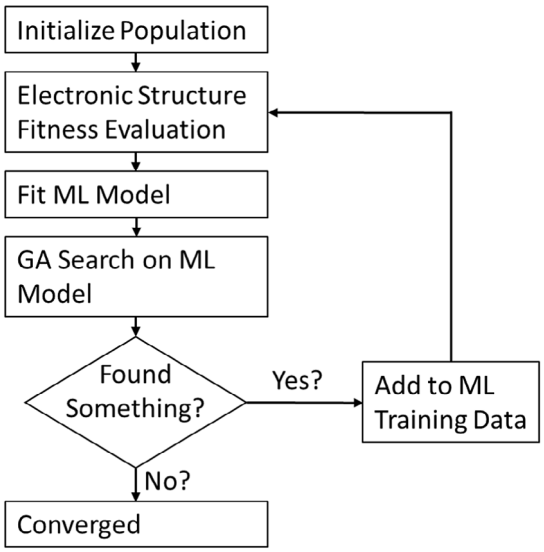


Figure 24. Schematic procedure of GA search on the basis of an ML-surrogate model. Reproduced under the terms of the CC BY 4.0 license.[72] Copyright 2019, Nature Publishing Group.

applied to select candidates with high fitness (low excess energy). A master GA worked on the basis of the demanding fitness evaluations and another nested GA acted on top of the fast surrogate model, which served as a high throughput screening function. In addition, Jennings et al. suggested further reduction in calculations might be possible using the candidates to further train the ML algorithm. Due to the serial nature involving the time demanding classical evaluation insufficient parallelizability might limit further improvements.[72]

As shown by Jennings et al. GAs can be used for the classification on the example of chemical candidates. As a next step, a GA like the one used by Finke et al.[57] could be used to develop a mathematical model based on the AI-modeled output, without requiring full human insight and ingenuity before obtaining the result. Furthermore, Jennings demonstrated the big potential to reduce modeling time by symbiotically use GAs in tandem with DL techniques. In addition, described methods like modular systems or complexity reduction via the technique of Dropout can additionally assist modeling.

7. Proposed Strategy: Procedure in Process Engineering Contexts

With the most relevant steps in the AI workflow described, an overall framework for AI applications with regard to the creation of predictive modeling as well as for the generation of new mechanistic models and insights can be derived, as shown in **Figure 25**.

In either case, a good understanding of the system to be modeled as well as the data source should be gained for the right selection of mechanisms and strategies regarding preprocessing, the selection of the AI tool, postprocessing techniques, etc. Data can be generated based on experiments and simulations or

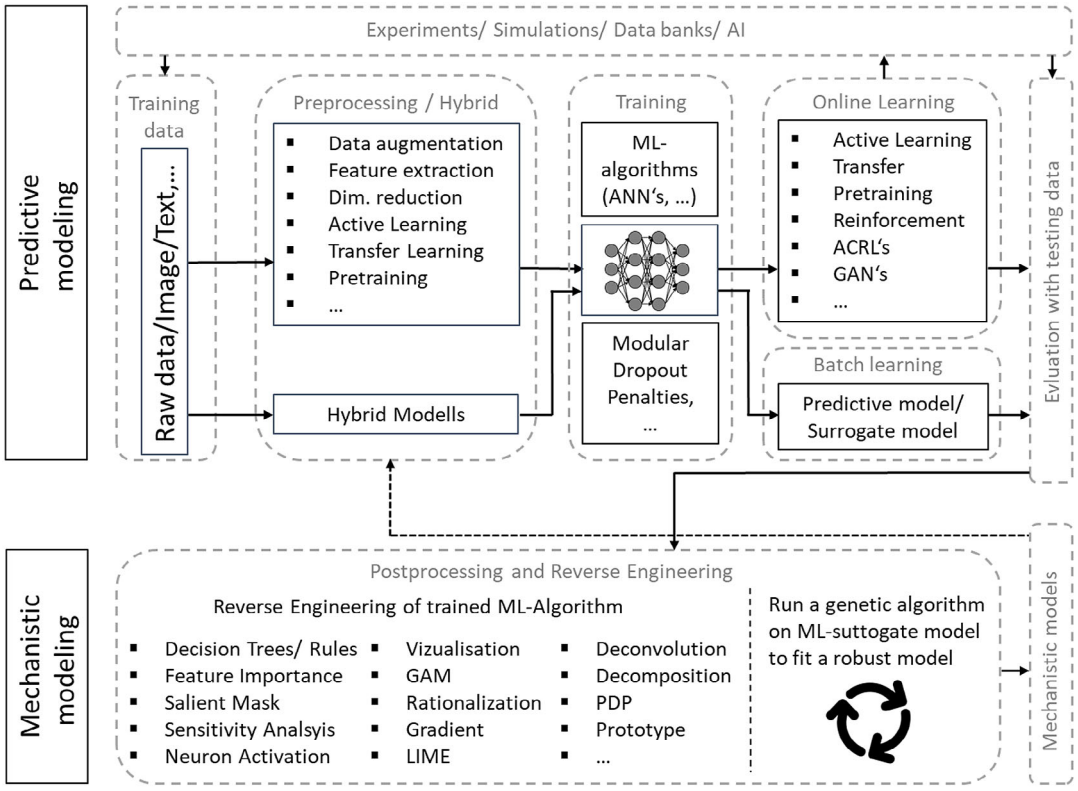


Figure 25. Proposed workflow for AI applications spitted to predictive and mechanistic modeling summarizing major techniques.

exported from existing databanks. The data can be tabulator data (numerical, categorical, or Boolean) as well as texts or pictures. Depending on the system to be studied, preprocessing strategies can be applied to facilitate the subsequent AI training. Some techniques like pretraining or Active Learning work best when conducted in a closed loop, also called Online Learning (in the contrast to Batch Learning), and therefore will play a role after the execution of the training phase. Especially useful in preprocessing is the application and inclusion of hybrid models by introducing pre-existing knowledge into the AI framework.

Afterward the actual training phase can be conducted. A multitude of AI techniques exist, of which some of the more important and often used techniques in mechanical- and process-engineering were described in this article. They range from statistical methods like regression algorithms, over GAs up to complex ML techniques like DL. Within the respective techniques, a multitude of possibilities exist to tailor the architecture of the algorithm toward the respective problem, like the application of modular neural net architectures.

The further process depends on whether the training is being applied in batch or online mode. In Batch Learning, the training occurs once on a fixed basis of data, whereas in Online Learning training is carried out in a closed loop with experiments, simulations, or sampling within a database. In the latter case, prominent examples are Reinforcement Learning and Active Learning. Other options are ACRL and GANs, in which the trained AI interacts with some sort of counterpart AI.

In either case, the trained AI is evaluated at the end or periodically for its accuracy and stability, usually by comparing its predictions with a portion of the data, which is not used for training the AI algorithm. This prevents evaluation with already overfitted data, which could lead to deceptively accurate evaluations. Usually, a proportion of 10–30% of the original data (in case of batch training) is set aside beforehand for this purpose. After reaching acceptable accuracy and stability, the model can be considered predictive. Depending on the inclusion of hybrid models, the thus obtained model is a black-box or gray-box model with little to no insights in its inner workings and mechanisms.

Successive steps can be applied to derive mechanistic models and insights. One approach comprises a multitude of methods with the goal to reverse engineer the inherent processes in the trained network. It aims to derive rules by basically breaking down the complexity of the network toward transparent and interpretable models. Preceding measures to reduce the networks complexity, like the application of Dropout, can facilitate the reverse engineering effort. Alternatively, the network can be run unaltered as a quick to apply surrogate model (in comparison with time-consuming experiments and simulations) for further modeling efforts, like the fitting of a mathematical model of limited complexity via a GA. As a consequence, mechanistic models can be derived, which may be integrated in another iteration step as additional white-box model in hybrid modeling to further facilitate the process of model building.

8. Perspective on Practical Applications

An overview of AI and its applications in a process engineering context has been given, together with a holistic strategy illustrating key steps and methods from a technical perceptive. When perusing to adopt the use of AI to a new practical process engineering usecase, be it for an academical or an industrial purpose, an important question to consider is how to introduce AI into the existing structures and workflows in an effective and efficient way to reduce starting friction and to quickly obtain first results. As an alternative to spending a lot of time and resources at the start into the setup of own hardware and own AI frameworks, the use of existing structures represents a quicker route to first results with the option to move to own solutions at a later state. The big tech companies most famous for their progress in AI offer rentable cloud solutions on highly specialized ASIC hardware and established frameworks, together with direct support to adapt the respective latest stage in AI to new applications. The biggest current providers of such platforms are Google Cloud,[73] Amazons AWS AI—Free AI Solutions,[74] Microsoft Azure AI,[75] or IBM's Watson.[76] In addition to always having access to the latest stage in AI, direct support from AI experts familiar with the respective tools and easy scalability of achieved solutions are great advantages. Moving from one platform to another has become a straightforward enterprise in recent years through tools like Google Cloud's Anthos.[77] In addition to hardware and frameworks the acquisition of knowledge regarding the AI techniques, software, usage etc. is required. In addition to preceding literature research, there are online and offline courses from many university's and companies as well as online learning platforms like Coursera, edX, Udacity, O'Reilly Online Learning, Lynda.com, or Fast.ai. Also the mentioned cloud providers often have projects for the education in AI, like Googles "Learn with Google AI" project.[78] In recent years, there has been a rise of platforms offering AI as a service (AIaaS), including the mentioned cloud providers Google, Amazon, Microsoft, and IBM. In AIaaS, the actual AI task can be outsourced to dedicated companies. Also crowd platforms exist, like Kaggle[79] or Experfy,[80] where a problem can be formulated to a crowd of data scientist. In these cases, it should however be evaluated that the published data sets are uncritical or normalized due to the wide distribution of the data. Looking specifically to the field of process engineering, there are already companies applying methods of AI in their operative business and companies focusing on building AI solutions for companies in process engineering. For instance, GE oil and BP apply an IoT-AI approach for predictive maintenance of their oil wells, connected to the internet, to reduce maintenance related downtimes and to optimize the profit margin as a result.[81] The company Uptake is focusing on building data analysis AI solutions and into integrating them in existing data sources and processes of client companies.[82] Similarly, the company Progress offers AI services to companies to use existing sensory data for anomaly detection.[83] In addition to individual companies, consortia of various institutions exist to gather companies, academia and governmental institutions to facilitate the adoption of AI techniques in industrial systems. Examples are the Consortium Project "Artificial Intelligence" set up by the Fraunhofer Institute for Production Technology IPT[84] or more domain-specific consortia like the "Machine Learning for Pharmaceutical Discovery and Synthesis Consortium" of the Massachusetts Institute of Technology.[85] Consulting a third-party company offering AI as a service or to blend into an existing consortium can facilitate the individual progress in terms of new

process engineering projects using methods of AI. Numerous examples exist for AI applications in industry and academia, they cover all of the mentioned main process engineering applications, namely predictive modeling, process optimization and control, and anomaly detection and mechanistic modeling. However, the scope of unexploited potential in all process engineering market segments currently still exceeds the number of existing applications by far. The most new applications to be expected in the near future are predictive modeling and anomaly detection related, due to their purely theoretical nature. They are followed by process optimization. Process control is more complex as it involves interactions with the physical world and is therefore related with a higher degree of necessary risk management. Likewise, mechanistic modeling is a more challenging for its subsequent position and the complexity caused by the preceding black box modeling. Nevertheless, there exist many tools and techniques such as hybrid modeling and reverse engineering strategies, which can relativize this rough classification (compare Section 5). It always depends on the respective application and the surrounding conditions and requires individual assessment. The developed workflow from Section 7 along with the reviewed methods can facilitate such assessment. There is no specific market segment which is more likely to be suitable for the application of AI then others. The increasing capability of AI is affecting all industries. It is unlikely that there is any industry not being affected or even disrupted by AI in the coming decade.

9. Conclusion

This study reviewed the current state and potential of the application of AI and more specifically ML in the field of process and chemical engineering. Starting with a brief introduction to the field of process and chemical engineering covering its respective fields of activity, tasks, and future potentials, the concept of AI was introduced. The article covers AIs' increasing capabilities driven by the developments in software, hardware, and the emergence of AI platforms. This demonstrated the increasing importance on a general level, as well as in the field of process and chemical engineering. Literature of already performed usecases of AI in the context of process and chemical engineering was examined. On this basis, a classification of the main tasks for AI in process and chemical engineering was derived. Individual exemplary cases were assigned to these tasks and edited according to the relevant details from a process engineering viewpoint as well as from an AI-specific viewpoint, acting as a roadmap for potential future perspectives in process and chemical engineering. In this context, an overview of the most important and most often used techniques and algorithms was provided, highlighting the most relevant details along with their respective advantages and disadvantages. In addition, all-encompassing steps in a typical workflow were identified, beginning with the data foundation, preprocessing strategies, the actual training of an AI, the use of hybrid models as well as postprocessing. Based on this, a holistic strategy or framework was presented, encompassing the relevant steps including potential applications related to the respective parts in the framework, with the aim of providing guidance in applying AI to novel process and chemical engineering projects, also a perspective on practical applications was provided. In the near future, AI will take an integral role in most—if not all—fields, including process and chemical engineering. Considering the exponential developments happening on multiple levels, which fuel each other, the possible applications in the near future will most likely far surpass the mentioned current applications, despite being already impressive at their current state. In the near future, the application of AI will enable new levels of process automation and process optimization. The prediction of highly complex production system behavior will be possible. Furthermore, such digital twins of entire plants could be designed automatically with little to no human intervention. Also, the discovery of mechanistic insights and models will be heavily facilitated. Over all, labor-intensive research procedures currently occupying years could be reduced to weeks or less, cutting costs, reducing resource consumption, and giving rise to entirely new applications in process and chemical engineering.

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

The authors want to thank the SMART BIOTECS alliance of Technische Universität Braunschweig and the Leibniz Universität Hannover. This initiative is supported by the Ministry of Science and Culture (MWK) of Lower Saxony, Germany.

Conflict of Interest

The authors declare no conflict of interest.

Keywords

artificial intelligence, hybrid modeling, mechanistic modeling, predictive modeling, process engineering

Received: November 20, 2020
Revised: January 13, 2021
Published online:

[1] X.-B. Jin, T.-L. Su, J.-L. Kong, Y.-T. Bai, B.-B. Miao, C. Dou, *Appl. Sci.* **2018**, *3*, 379.
[2] J.-P. Dal Pont, *Process Engineering and Industrial Management*, Wiley, London **2013**.
[3] I. E. Grossmann, A. W. Westerberg, *AIChE J.* **2000**, *9*, 1700.
[4] P. Ongsulee, in *ICT-KE IC, Proc. 2017 Fifteenth Int. Conf. on ICT and Knowledge Engineering*, IEEE, Piscataway, NJ **2017**, pp. 1–6.
[5] L. Deng, *FNT Signal Proc.* **2014**, *34*, 197.
[6] V. A. Golovko, *Opt. Mem. Neural Netw.* **2017**, *26*, 1.
[7] G. E. Hinton, *Neural Comput* **2002**, *8*, 1771.
[8] G. E. Hinton, R. R. Salakhutdinov, *Science* **2006**, *5786*, 504.
[9] G. E. Hinton, S. Osindero, Y.-W. Teh, *Neural Comput.* **2006**, *7*, 1527.
[10] G. E. Moore. *IEEE Solid-State Circuits Soc. Newslett.* **2006**, *3*, 33.
[11] R. Kurzweil, *The Singularity Is Near*, Viking, New York **2005**.

[12] M. van den Brink, in *2019 IEEE Int. Electron Devices Meeting (IEDM)*, IEEE, Piscataway, NJ **2019**, pp. 1.2.1–1.2.5.
[13] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, et al., *Nature* **2019**, *7779*, 505.
[14] NVIDIA Tesla V100, **2020**, https://www.nvidia.com/de-de/data-center/tesla-v100/ (accessed: 02 April 2020).
[15] Google Cloud TPU, **2020**, https://cloud.google.com/tpu?hl=de (accessed: 02 April 2020).
[16] Cerebras, **2020**, https://www.cerebras.net/ (accessed: 02 April 2020).
[17] D. Amodei, OpenAI **2018**, https://openai.com/blog/ai-and-compute/.
[18] D. Hernandez, T. B. Brown, *Measuring the Algorithmic Efficiency of Neural Networks*, **2020**, arXiv:2005.04305v1.
[19] R. E. Bixby, *Documenta Mathematica*, Extra Volume ISMP, Deutsche Mathematiker-Vereinigung **2012**, pp. 107–121.
[20] K. Grace, Algorithmic Progress in Six Domains. Tech. rep. Machine Intelligence Research Institute **2013**, http://intelligence.org/files/AlgorithmicProgress.pdf.
[21] Z. Wang, K. Liu, J. Li, Y. Zhu, Y. Zhang, *Arch Computat Methods Eng* **2019**, https://doi.org/10.1007/s11831-018-09312-w.
[22] A. Géron, *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow*, O'Reilly Media, Inc, Sebastopol, CA **2019**.
[23] J. Brownlee, *Deep Learning With Python: Develop Deep Learning Models on Theano and TensorFlow Using Keras*, Machine Learning Mastery, Melbourne **2016**.
[24] H. Jin, Q. Song, X. Hu, *Auto-Keras: An Efficient Neural Architecture Search System, in: Proceedings of the 25th ACM*, SIGKDD International Conference on Knowledge Discovery & Data Mining, ACM **2018**, pp. 1946–1956.
[25] Z. Geng, Q. Meng, Y. Han, Q. Wei, Z. Ouyang, in *Proc. of 2019 IEEE 8th Data Driven Control and Learning Systems Conf. (DDCLS'19)*, IEEE, Piscataway, NJ **2019**, pp. 708–712.
[26] Z. Zhang, Z. Wu, D. Rincon, P. D. Christofides, *Mathematics* **2019**, *10*, 890.
[27] J. Günther, P. M. Pilarski, G. Helfrich, H. Shen, K. Diepold, *Mechatronics* **2016**, *34*, 1.
[28] S. Shastri, C.-P. Lam, B. Werner, *J. Chem. Eng. Japan/JCEJ* **2004**, *6*, 691.
[29] A. Sokolov, I. Pyatnitsky, S. Alabugin, *FME Trans.* **2019**, *4*, 782.
[30] Q. Xing-yu, Z. Peng, X. Chengcheng, F. Dong-dong, in *IEEE-CYBER 2017: The 7th Annual IEEE Int. Conf. on Cyber Technology in Automation, Control and Intelligent Systems*, IEEE, Piscataway, NJ **2017**, pp. 673–678.
[31] A. Dey, *Int. J. Comput. Sci. Inf. Technol.* **2016**, *3*, 1174.
[32] J. Holloway, K. Mengersen, *Remote Sens.* **2018**, *9*, 1365.
[33] D. Reinsel, J. Gantz, J. Rydning, *IDC White Paper*, **2018**.
[34] A. Mikolajczyk, M. Grochowski, in *2018 Int. Interdisciplinary PhD Workshop (IIPhDW)*, IEEE, Piscataway, NJ **2018**, pp. 117–122.
[35] Y. Bengio, A. Courville, P. Vincent, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2012**, *35*, 1798.
[36] V. Samsonov, J. Lipp, P. Noodt, A. F. Solvay, T. Meisen, in *2019 IEEE Symp. Series on Computational Intelligence (SSCI)*, IEEE, Piscataway, NJ **2019**, pp. 799–807.
[37] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, *J. Mach. Learn. Res.* **2014**, *1*, 1929.
[38] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, D. Hassabis, *Nature* **2017**, *7676*, 354.
[39] M. Kehrer, J. Pitz, T. Rothermel, H.-C. Reuss, in *Internationales Stuttgarter Symp.: Automobil- und Motorentechnik* (Eds: M. Bargende, H.-C. Reuss, J. Wiedemann). Springer Vieweg, Wiesbaden, Germany **2018**, pp. 659–669.
[40] Waymo, **2018**, Waymo Safety Report. On the Road to Fully Self-Driving, https://storage.googleapis.com/sdc-prod/v1/safety-report/Safety%20Report%202018.pdf (accessed: 09 April 2020).
[41] D. Pfau, O. Vinyals **2016**, arXiv:1610.01945v3.
[42] P. Korshunov, S. Marcel, *DeepFakes: a New Threat to Face Recognition? Assessment and Detection*, **2018**, arXiv:1812.08685v1.
[43] D. Li, D. Chen, J. Goh, S.-K. Ng, *Anomaly Detection with Generative Adversarial Networks for Multivariate Time Series*, **2018**, arXiv:1809.04758v3.
[44] A. Antoniou, A. Storkey, H. Edwards, *Data Augmentation Generative Adversarial Networks*, **2017**, arXiv:1711.04340v3.
[45] S. J. Pan, Q. Yang, *IEEE Trans. Knowl. Data Eng.* **2010**, *10*, 1345.
[46] X. Yuan, C. Ou, Y. Wang, C. Yang, W. Gui, *Chem. Eng. Sci.* **2020**, *217*, 115509.
[47] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, H. Arshad, *Heliyon* **2018**, *11*, e00938.
[48] M. Olson, A. Wyner, R. Berk, in *Advances in Neural Information Processing Systems 31* (Eds: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett), Curran Associates, Inc, Red Hook **2018**, pp. 3619–3628.
[49] D. C. Psichogios, L. H. Ungar, *AIChE J* **1992**, *10*, 1499.
[50] B. Kiranmai, A. Damodaram, *Int. J. Eng. Comput. Sci.* **2014**, *3*, 7217.
[51] M. L. Thompson, M. A. Kramer, *AIChE J* **1994**, *8*, 1328.
[52] S. Zendehboudi, N. Rezaei, A. Lohi, *Appl. Energy* **2018**, *228*, 2539.
[53] A. Sundaram, P. Ghosh, J. M. Caruthers, V. Venkatasubramanian, *AIChE J.* **2001**, *6*, 1387.
[54] S. Curcio, A. Saraceno, V. Calabrò, G. Iorio, A. Tariq, A. Bekatorou, A.-U. Amin, *Sci. World J.* **2014**, *2014*, 303858.
[55] M. Tie, H. Yue, T. Chai, in *Advances in Neural Networks – ISNN 2005*, Vol. *3498* (Eds: X. Liao, J. Wang, Z. Yi), Springer-Verlag, Berlin/Heidelberg **2005**, pp. 871–876.
[56] N. Bhutani, G. P. Rangaiah, A. K. Ray, *Ind. Eng. Chem. Res.* **2006**, *23*, 7807.
[57] B. Finke, C. Sangrós Giménez, A. Kwade, C. Schilde, unpublished.
[58] R. Kurzweil, *How to Create a Mind*, Penguin Books, New York, NY **2013**.
[59] A. S. Morcos, D. G. T. Barrett, N. C. Rabinowitz, M. Botvinick, *On the Importance of Single Directions for Generalization*, **2018**.
[60] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, D. Pedreschi, *ACM Comput. Surv.* **2019**, *5*, 1.
[61] V. Buhrmester, D. Münch, M. Arens, *Analysis of Explainers of Black Box Deep Neural Networks for Computer Vision: A Survey*, **2019**, arXiv:1911.12116v1.
[62] Y. Lou, R. Caruana, J. Gehrke, in *Proc. of the 18th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, ACM, New York, NY **2012**, p. 150.
[63] U. Ehsan, B. Harrison, L. Chan, M. O. Riedl, *AIES'18: Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society* **2018**, pp. 81–87.
[64] D. H. Park, L. A. Hendricks, Z. Akata, B. Schiele, T. Darrell, M. Rohrbach, *Attentive Explanations: Justifying Decisions and Pointing to the Evidence*, **2016**, arXiv:1612.04757v2.
[65] P. Voosen, *Science* **2017**, https://doi.org/10.1126/science.aan7059.
[66] K. Simonyan, A. Vedaldi, A. Zisserman, *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*, **2013**, arXiv:1312.6034v2.
[67] C. Mark, J. W. Shavlik, in *Advances in Neural Information Processing Systems*, **1996**, pp. 24–30.

[68] C. Mark, J. W. Shavlik, in *Proc. of the Eleventh Int. Conf. on Machine Learning*, **1994**.
[69] A. Binder, G. Montavon, S. Bach, K.-R. Müller, W. Samek, *Layer-Wise Relevance Propagation for Neural Networks with Local Renormalization Layers*, **2016**, https://arxiv.org/pdf/1604.00825.
[70] J. Bien, R. Tibshirani, *Ann. Appl. Stat.* **2011**, *4*, 2403.
[71] G. Hooker, in *KDD-2004: Proc. of the Tenth ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, ACM Press, New York, NY **2004**, p. 575.
[72] P. C. Jennings, S. Lysgaard, J. S. Hummelshøj, T. Vegge, T. Bligaard, *npj Comput. Mater.* **2019**, *5*, 46.
[73] Google Cloud, 2021.000Z, Google Cloud Computing, Hosting Services & APIs, https://cloud.google.com/products/ai (accessed: 10 January 2021).
[74] I. Amazon Web Services, 2020.000Z, Machine Learning for Free, https://aws.amazon.com/free/machine-learning/ (accessed: 10 January 2021).
[75] AI Platform | Microsoft Azure, https://azure.microsoft.com/en-us/overview/ai-platform/, (accessed 10 January 2021).
[76] IBM Watson products, **2020**, https://www.ibm.com/watson/products-services (accessed: 10 January 2021).
[77] Google Cloud, 2021.000Z, Google Cloud Anthos, https://cloud.google.com/anthos, (accessed: 10 January 2021).
[78] Google AI, 2021.000Z, Education – Google AI, https://ai.google/education (accessed: 10 January 2021).
[79] Kaggle: Your Machine Learning and Data Science Community, https://www.kaggle.com/ (accessed: 10 January 2021).
[80] Analytics Consulting | AI Products | Big Data Consulting | AI Consulting, https://www.experfy.com/ (accessed: 10 January 2021).
[81] Deep Machine Learning: GE and BP Will Connect Thousands of Subsea Oil Wells to the Industrial Internet | GE News, https://www.ge.com/news/reports/deep-machine-learning-ge-and-bp-will-connect-2 (accessed: 10 January 2021).
[82] V. Venkatasubramanian, *AIChE J.* **2019**, *2*, 466.
[83] Progress.com, How Cognitive Anomaly Detection and Prediction Works – Progress DataRPM, https://www.progress.com/datarpm/how-it-works (accessed 10 January 2021).
[84] Fraunhofer Institute for Production Technology IPT, Consortium Project "Artificial Intelligence" – Fraunhofer IPT, https://www.ipt.fraunhofer.de/en/Competencies/Technologymanagement/consortium-projects/artificial-intelligence.html (accessed: 10 January 2021).
[85] MLPDS – Machine Learning for Pharmaceutical Discovery and Synthesis Consortium, https://mlpds.mit.edu/ (accessed: 10 January 2021).
[86] Carnegie Mcellon University: The robotics institute, MIPS Equivalents, https://frc.ri.cmu.edu/~hpm/book97/ch3/processor.list (accessed: 15 July 2020).
[87] MMID, 2019, IoT: Why Now? – MMID, https://mmid-group.com/iot-why-now/ (accessed: 15 July 2020).
[88] The Median Group, 2019.000Z, How rapidly are GPUs improving in price performance? – Median Group, http://mediangroup.org/gpu.html (accessed: 15 Jul 2020).
[89] C. Halfmann, H. Holzmann, *Adaptive Modelle für die Kraftfahrzeugdynamik*, Springer, Berlin, Heidelberg **2013**.
[90] D. Patel, V. Vakharia, M. Kiran, *FME Trans.* **2019**, *4*, 865.
[91] A. V. Kalyuzhnyuk, R. L. Lapin, A. S. Murachev, A. E. Osokina, A. I. Sevostianov, D. V. Tsvetkov, *Mater. Phys. Mech.* **2019**, *42*, 351.
[92] W. A. Armansyah, J. Saedon, *J. Mech. Eng.* **2018**, *5*, 216.
[93] F. Rosenblatt, *Psychol. Rev.* **1958**, *6*, 386.
[94] A. G. Ivachnenko, V. G. Lapa, *Cybernetics and Forecasting Techniques*, American Elsevier, New York **1967**.
[95] A. Krizhevsky, I. Sutskever, G. E. Hinton, *Commun. ACM* **2017**, *6*, 84.
[96] J. Elman, *Cognitive Science* **1990**, *2*, 179.
[97] S. Hochreiter, J. Schmidhuber, *Neural Comput.* **1997**, *8*, 1735.
[98] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, *Proc. IEEE* **1998**, *11*, 2278.
[99] S. Fine, Y. Singer, N. Tishby, *Machine Learning* **1998**, *1*, 41.
[100] C. Cortes, V. Vapnik, *Machine Learning* **1995**, *3*, 273.
[101] A. S. Pensado, A. A. H. Pádua, *Angew. Chemie* **2011**, *37*, 8683.

Christoph Thon studied bio- and chemical engineering at the Technische Universität Braunschweig and the Ostfalia University of Applied Sciences from 2011 to 2017 and finished with his master thesis "Investigation of the stress distribution through laminar flows by using CFD for scale transfer of kneading processes". His doctoral research under supervision by Prof. Dr.-Ing. Carsten Schilde at the Institute for Particle Technology (iPAT) focuses on the investigation of grinding processes in mills by means of computer simulations and the application of Artificial Intelligence.

Benedikt Finke studied bioengineering at the Technische Universität Braunschweig 2007–2013 and finished with his diploma thesis "Influence of the dispersing process on the particle size in epoxy resins and the mechanical properties of nanocomposites". His doctoral research under supervision by Prof. Dr.-Ing. Arno Kwade at the Institute for Particle Technology (iPAT) focuses on the modeling of rheological properties of nanoparticulate suspensions and the modeling of dispersing processes.

Arno Kwade studied mechanical engineering at TU Braunschweig and University of Waterloo, finishing his Ph.D. thesis in 1996. He is the head of Institute for Particle Technology, scientific director of the Battery-Lab-Factory, and spokesman of the Center-of-Pharmaceutical-Engineering at TU Braunschweig. He is division head of Fraunhofer Project Center for Energy Storage and Systems, member of the National Academy of Science and Engineering, chairman of the ProcessNet and EFCE working parties "Comminution and classification," vice chairman of the Advisory Board Battery Research Germany and chair of the PARTEC 2022.

Carsten Schilde studied bioengineering at the Technische Universität Braunschweig and completed his Ph.D. thesis about "Structure, Mechanics and Fracture of Nanoparticulate Aggregates" under Prof. Dr.-Ing. Arno Kwade. Later he worked at the Institute for Particle Technology as research associate and was appointed assistant professor for "Particle Simulation & Functional Structures" 2017. His research focuses on the investigation of particulate systems and structures using Computer Simulations and AI. Main application fields are pharmaceuticals and composite materials.